

# Classifications and canonical forms of tensor product expressions in the presence of permutation symmetries

Zhendong Li,<sup>1 a) b)</sup> Sihong Shao,<sup>2 c)</sup> and Wenjian Liu<sup>1 d)1</sup>

<sup>1</sup> *Beijing National Laboratory for Molecular Sciences, Institute of Theoretical and Computational Chemistry, State Key Laboratory of Rare Earth Materials Chemistry and Applications, College of Chemistry and Molecular Engineering, and Center for Computational Science and Engineering, Peking University, Beijing 100871, People's Republic of China*

<sup>2</sup> *LMAM and School of Mathematical Sciences, Peking University, Beijing 100871, People's Republic of China.*

Complicated mathematical equations involving tensors with permutation symmetries are frequently encountered in fields such as quantum chemistry, e.g., those in coupled cluster theories and derivatives of wavefunction parameters. In automatic derivations of these equations, a key step is the collection of product terms that can be found identical by using permutation symmetries or relabelling dummy indices. In the present work, we define a canonical form for a general tensor product in the presence of permutation symmetries as a result of *the classification of all tensor products from a group theoretical point of view*. To make such definition of practical use, we provide an efficient algorithm to compute the canonical form by combining the classical backtrack search for permutation groups and the idea of partitions used in graph isomorphism algorithms. The resulted algorithm can compute canonical forms and generators of the automorphism groups of tensor expressions. Moreover, for tensor products with external indices, its permutation group can be determined via the quotient group of the automorphism group for the corresponding *externally and internally unlabeled graph (skeleton)* with respect to that for the *externally labeled and internally unlabeled graph (diagram)*.

---

<sup>a)</sup> To whom correspondence should be addressed: zhendongli2008@gmail.com

<sup>b)</sup> Present address: Department of Chemistry, Princeton University, Princeton, New Jersey 08544, USA

<sup>c)</sup> To whom correspondence should be addressed: sihong@math.pku.edu.cn

<sup>d)</sup> To whom correspondence should be addressed: liuwj@pku.edu.cn

# CONTENTS

<b>I. Introduction</b>	2
<b>II. Theory</b>	4
A. Classification of tensor products	5
B. Examples	12
<b>III. Algorithms</b>	14
A. Traditional backtrack algorithm	14
B. Graphic representation of tensor products	16
C. Partition backtrack algorithm	19
1. Search tree	20
2. Pruning with non-discrete partition	22
3. Pruning with automorphism	23
<b>IV. Results and discussions</b>	26
<b>V. Conclusion and outlook</b>	29
<b>References</b>	35

## I. INTRODUCTION

In its most broad sense, tensors are multi-dimensional arrays of numerical values. They are ubiquitous in quantum chemistry ranging from two-electron integrals to amplitudes of excited configurations in electronic structure models<sup>1</sup>. For instance, a general  $n$ -tuple excitation from a reference  $|\Phi_0\rangle$  can be expressed as  $|\Psi_{n\text{-tuple}}\rangle = \sum_{\{p_i\}, \{h_i\}} t_{h_1 h_2 \dots h_n}^{p_1 p_2 \dots p_n} a_{h_1 h_2 \dots h_n}^{p_1 p_2 \dots p_n} |\Phi_0\rangle$ , where  $t_{h_1 h_2 \dots h_n}^{p_1 p_2 \dots p_n}$  is a  $2n$ -way tensor. One of the key challenges of quantum chemistry is to develop efficient algorithms to deal with the optimization of wavefunction parameters and evaluation of energy derivatives for molecular properties. Unfortunately, as the theory becomes more and more sophisticated, the manual manipulation of the resulted equations becomes difficult and automatic derivations need to be used. A key step in such automatic derivations is the collection of product terms that are identical by using permutation

symmetries or relabelling dummy indices. In this paper, we examine some important questions about the permutation symmetry of tensor products. Specifically, let a general tensor product be denoted by the form

$$X_1^{E_1 I_1} X_2^{E_2 I_2} \dots X_k^{E_k I_k}, \quad (1)$$

where the tensor  $\mathbf{X}_i$  represents the  $i$ -th factor,  $E_1 \cup E_2 \cup \dots \cup E_k = E$  form a disjoint partition of an external index set  $E$ ,  $I_i$  is the set of internal indices that will be contracted, and hence  $I_i \cap I_j$  represent the set of internal indices to be contracted between  $\mathbf{X}_i$  and  $\mathbf{X}_j$  or more explicitly  $X_i^{E_i I_i}$  and  $X_j^{E_j I_j}$  in component form. Suppose the tensors  $\mathbf{X}_i$  possess some index permutation symmetry, we want to know the answers for the following four closely related questions

- (i) Whether two given tensor expressions of the form (1) are identical? If not, to what extent they are different?
- (ii) How many different tensors can be formed from a same set of factors? For a given member, how many of products are equivalent (numerically identical) to it?
- (iii) Among these equivalent tensors, is it possible to select a unique representative?
- (iv) Let the tensor obtained in Eq. (1) be denoted by  $Z^E$ , then what is its permutation symmetry?

These fundamental questions must be faced when designing general automatic derivation and simplification tools based on algebraic expressions. Without symmetry, these questions are trivial. However, with symmetry they may become complicated, since the possible algebraic forms of products in Eq. (1) can be enormous.

In automating the derivation of the high-spin open-shell coupled cluster singles and doubles (CCSD), Janssen and Schaefer<sup>2</sup> defined a "canonical" form for a tensor product by comparing the permuted index arrays obtained via all possible index permutations of the factors  $\mathbf{X}_i$ . In this way, two equivalent terms become identical and can be rapidly combined. Such exhaustive procedure is complete (guaranteed to find the optimum), but becomes impractical if the term presented is of form  $T_k^n$  (cluster amplitude) with large  $k$  or  $n$  in view of the rapid increase of the factorial. Latter, the tensor contraction engine (TCE) developed by Hirata et al.<sup>3</sup> followed the similar idea, but in order to treat high-order tensors, in which the

above exhausting permutations of all indices become formidable, some kinds of sorting based on several predefined rules are used. However, for some "cyclic" tensor like  $(t_{h_3 h_4}^{p_1 p_2})^* t_{h_3 h_4}^{p_5 p_6} t_{p_5 p_6}^{p_1 p_2}$ , a comparison by permuting all common indices have to be carried out because the sorting of tensors and the sorting of indices are intertwined. Thus, we should design more economic approaches. Such approaches should utilize the properties of the "canonical" functions, rather than blindly searching all possibilities. An example is that for a cluster amplitude  $T_n$ , if the canonical form is defined by the ordering of indices, then the problem is simply solved by a sorting procedure, which can obviously be achieved with a cost at most  $O(n^2)$  instead of  $O(n!)$ . The definition of canonical form is in some sense arbitrary, because it depends on some predefined criteria. However, the canonical form should at least satisfies two conditions. First, it should be universal which means it can be applied to any kind of product in Eq. (1). Second, in order to make such form of practical use, there should be efficient algorithms to compute it.

In this work, we answered the above questions (i)-(iv) by using group and graph theoretical approaches. In Sec. II, we provide a classification of all tensor products by introducing five equivalence relations. In Sec. III, in order to establish an efficient algorithm to compute the canonical form defined in Sec. II, an elegant mapping from algebraic tensor product to graph is defined. By combining the classical backtrack search for permutation groups and the idea of partitions from graph isomorphism algorithms, we provide an efficient algorithm to compute the canonical form of a tensor product and the automorphism group of its associated graph. The latter provides detailed information about the permutation symmetry of the resulted tensor. Compared to the Butler-Portugal algorithm<sup>4-7</sup> for index canonicalization with respect to permutation symmetries based on finding the doublet coset representatives<sup>8</sup>, the present algorithm utilized the concepts of equivalence relation, colorings, and group chains<sup>9-12</sup>, as well as some graphical ideas and the related graph isomorphism algorithms<sup>13-16</sup>. Sec. IV shows some results for several tensor products presented in quantum chemistry. The conclusion and outlook are presented in Sec. V.

## II. THEORY

The basic tool we used for classification is the concept of equivalence relation.

**Definition 1** (equivalence relation). *A given binary relation  $\sim$  on a set  $M$  is said to be an*

equivalence relation if and only if it satisfies three requirements: (1) (reflexivity)  $a \sim a$ , (2) (symmetry) if  $a \sim b$  then  $b \sim a$ , (3) (transitivity) if  $a \sim b$  and  $b \sim c$  then  $a \sim c$ . The equivalence relation partitions the set  $M$  into disjoint equivalence classes  $[x]$ , which are defined via  $[x] = \{x' \in M | x' \sim x\}$ . For two equivalence relations  $\sim$  and  $\approx$  defined on the same set  $X$ , and  $a \approx b$  implies  $a \sim b$  for all  $a, b \in X$ , then  $\sim$  is said to be a coarser relation than  $\approx$ , and  $\approx$  is a finer relation than  $\sim$ .

The permutation symmetry of a tensor can be characterized by its associated permutation group defined in the following way.

**Definition 2** (tensor symmetry). For a  $r$ -way tensor  $\mathbf{Z}$ , the set of permutations satisfying the condition

$$g_i \circ \mathbf{Z}^P \triangleq \mathbf{Z}^{g_i(P)} = \mathbf{Z}^P, \quad g_i \in \mathcal{S}_r, \quad (2)$$

where  $\mathcal{S}_r$  is the symmetric group of degree  $r$ , and  $P$  is a set of abstract indices,

$$g_i(P) = g_i(p_1 p_2 \cdots p_r) = p_{g_i(1)} p_{g_i(2)} \cdots p_{g_i(r)} \quad (3)$$

forms a permutation group under the composition of permutations. We refer to it as the permutation symmetry group of the tensor  $\mathbf{Z}$ , denoted by  $\mathcal{G}(\mathbf{Z})$ .

The meaning of Eq. (2) is transparent. It reveals that the elements of  $\mathbf{Z}$  are related in some way. It can be viewed as an extension of the transpositional symmetry of matrix, in which case we can have symmetric and antisymmetric matrices  $(12) \circ A_{p_1 p_2} = A_{p_2 p_1} = \pm A_{p_1 p_2}$ . Note that the antisymmetry is not covered by Eq. (2). In principle, we can exploit more symmetry in  $\mathbf{Z}$ , e.g., by considering the permutations whose actions only change the phase of the tensors. Such case can be incorporated in our framework introduced below by defining more general permutation symmetry groups. However, for the sake of simplicity, in the following discussions, we only consider the symmetry defined in Eq. (2).

## A. Classification of tensor products

First of all, let us make our targets (i)-(iv) clear by defining them more precisely.

**Definition 3** (symmetry equivalent tensors). Two tensor products  $\mathbf{Z}_1$  and  $\mathbf{Z}_2$  of form (1) are symmetry equivalent, if there exist a permutation  $g$  of factors and indices such that  $\mathbf{Z}_1 = g \circ \mathbf{Z}_2$ . In this case, we denote them by  $\mathbf{Z}_1 \sim \mathbf{Z}_2$ .

By this definition, the first necessary condition for  $\mathbf{Z}_1 \sim \mathbf{Z}_2$  is that they must share the same set of factors, otherwise, even when neglecting the indices it is not possible to match them by rearrangement of factors. Let us denote this factor set by  $S = \{\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_k\}$ . We denote an ordered set by  $s_i = (\mathbf{X}_{i_1}, \mathbf{X}_{i_2}, \dots, \mathbf{X}_{i_k})$  where  $i_1 i_2 \dots i_k$  is a permutation of  $\{1, 2, \dots, k\}$ . It is always possible to define an order for all possible orderings of  $S$ . For instance, we can use the simple lexicographical order, namely, for two different orderings of  $S$ ,  $s_1 = (\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_k)$  and  $s'_1 = (\mathbf{X}'_1, \mathbf{X}'_2, \dots, \mathbf{X}'_k)$ , we say  $s_1 < s_2$  if and only if the first  $\mathbf{X}_i$ , which is different from  $\mathbf{X}'_i$ , comes before  $\mathbf{X}'_i$  in the alphabet. From now on, we assume  $S$  is lexicographically ordered.

**Definition 4.** (*factor set and the associated permutation group*) Given the factor set  $S = (\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_k)$  and the permutation symmetry group of each factor  $\mathcal{G}(\mathbf{X}_i)$ , we can define the symmetry group of  $S$  via direct product, viz.,  $\mathcal{G}(S) = \mathcal{G}(\mathbf{X}_1) \times \mathcal{G}(\mathbf{X}_2) \times \dots \times \mathcal{G}(\mathbf{X}_k)$ . If some factors are the same, e.g.,  $\mathbf{X}_i = \mathbf{X}_{i+1} = \dots = \mathbf{X}_{i+n-1}$ , then the corresponding parts of direct product  $\mathcal{G}_n(\mathbf{X}_i) \triangleq \mathcal{G}(\mathbf{X}_i) \times \mathcal{G}(\mathbf{X}_i) \times \dots \times \mathcal{G}(\mathbf{X}_i)$  should be replaced by the semidirect product  $\mathcal{G}_n(\mathbf{X}_i) \rtimes \mathcal{S}_n(\mathbf{X}_i)$ , where  $\mathcal{S}_n(\mathbf{X}_i)$ , which is isomorphic to  $S_n$ , represents the symmetric group for permutations of the identical factors.

We can label each dimension of  $S$  by a consecutive integer number. The action of  $\mathcal{G}(S)$  on  $S$  naturally induces an action on the integer set  $\Omega = \{1, 2, \dots, D\}$  where  $D = \sum_{i=1}^k d_i$  with  $d_i$  being the dimension of the tensor  $\mathbf{X}_i$ . For simplicity, as  $\mathcal{G}(S)$  of  $S$  and the induced permutation group of  $\Omega$  are isomorphic, we do not distinguish them and denote both groups by  $\mathcal{G}(S)$ . Now we focus on the index structure of tensor products. According to Eq. (1), the set of indices can be denoted by  $\pi(\Omega) = E_1 I_1 \cup E_2 I_2 \cup \dots \cup E_k I_k \triangleq \{\pi(1), \pi(2), \dots, \pi(D)\}$ . From this, we can see that  $\pi$  can be viewed as a *coloring* of  $\Omega$ , a mapping from  $\Omega$  to a color set, which the set of indices played a role as. Then we can establish the following connection.

**Theorem 1.** *The classification of different tensor products (1) with the same factor set  $S$  under the permutation symmetry group  $\mathcal{G}(S)$  is equivalent to the classification of different colorings  $\pi$  of  $\Omega$  under the induced permutation group on  $\Omega$ .*

In this work, we are interested in the tensor products that each internal index appear only twice. This is usually the case as required by the invariance of equations under orbital rotations. For simplicity, we will use the Einstein summation convention for tensor products.

Because the internal indices in tensor products are free to be permuted (relabelled) without changing the final result, we have to introduce a group  $\mathcal{H}$  to describe the invariance for permutation of colorings. The group  $\mathcal{H}$  is determined from the types of internal indices, e.g., occupied or virtual, and only the internal indices of the same type are allowed to be permuted. For given number of external indices  $n_{ext}$  and contracted internal pairs  $n_c$ , then  $n_{int} = 2n_c$ ,  $D = n_{ext} + n_{int}$ , and the number of colorings  $C = n_{ext} + n_c$ . There are totally  $N(n_{ext}, n_{int}) = D!/2^{n_c}$  different expressions of form (1). However, most of them represent the same result. This point is formalized by the following equivalence relation.

**Definition 5.** *Two colorings  $\pi_1$  and  $\pi_2$  are equivalent if and only if there exist  $g \in \mathcal{G}(S)$  and  $h \in \mathcal{H}$  such that  $\pi_1 g = h \pi_2$ . For simplicity, we refer this equivalence relation as  $(\mathcal{G}(S), \mathcal{H})$ -equivalence.*

This definition essentially characterizes the Definition 3 for tensor products (1). Under this equivalence relation, the  $N(n_{ext}, n_{int})$  different expressions can be classified into equivalent classes. The enumeration of nonequivalent colorings in the presence of permutation symmetries  $\mathcal{G}(S)$  and  $\mathcal{H}$  is a classical combinatorial problem that is solved by de Bruijn's generalization<sup>11,12</sup> of the Pólya-Redfield theory<sup>9,10</sup>.

**Lemma 1** (de Bruijn's enumeration formula). *Suppose  $\mathcal{Y} = \{y_1, \dots, y_C\}$  is a set of colors, and  $\mathcal{H}$  is a subgroup of  $S_C$ , then the generating function for the colorings of  $\Omega$  which are nonequivalent with respect to the action of  $\mathcal{G}(S)$  on  $\Omega$  and the action of  $\rho$  on  $\mathcal{Y}$  can be obtained by identifying equivalent color patterns in the polynomial*

$$F_{\mathcal{G}, \mathcal{H}}(y) = \frac{1}{|\mathcal{H}|} \sum_{h \in \mathcal{H}} P_{\mathcal{G}}(\alpha_1(h), \alpha_2(h), \dots, \alpha_D(h)) \quad (4)$$

where  $P_{\mathcal{G}}(x_1, x_2, \dots, x_D)$  is the cycle index of  $\mathcal{G}$  defined by

$$P_{\mathcal{G}}(x_1, x_2, \dots, x_D) = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \prod_{i=1}^n x_{l_i} \quad (5)$$

with  $g$  being a product of  $n$  cycles, and the  $i$ th cycle has length  $l_i$ . The  $\alpha_m(h)$  is defined by

$$\alpha_m(h) = \sum_{\{j : h^m(j)=j, 1 \leq j \leq C\}} \prod_{i=0}^{m-1} y_{h^i(j)}, \quad (6)$$

for  $1 \leq m \leq D$ .

Let the first  $n_{ext}$  elements of  $\mathcal{Y}$  correspond to external indices, while the remaining correspond to internal indices, by using Lemma 1 one can find the number of nonequivalent tensor products with respect to  $\mathcal{G}(S)$  from the coefficient of the monomial  $y_1 y_2 \cdots y_{n_{ext}} y_{n_{ext}+1}^2 y_{n_{ext}+2}^2 \cdots y_C^2$ . By setting  $\mathcal{H} = \{e\}$ , Lemma 1 reduces to the Pólya's theorem<sup>9</sup> and then Eq. (4) gives the number of nonequivalent classes under the equivalence relation  $\pi_1 = \pi_2 g$  with  $g \in \mathcal{G}(S)$  for  $\pi_1$  and  $\pi_2$ . We refer this equivalence relation as  $\mathcal{G}(S)$ -equivalence. For a coloring  $\pi$ , its  $\mathcal{G}(S)$ -equivalent class is denoted by  $\bar{\pi} = \{g \circ \pi(\Omega) : g \in \mathcal{G}(S)\}$ , and the  $(\mathcal{G}(S), \mathcal{H})$ -equivalent class is denoted by  $\bar{\bar{\pi}} = \{h \circ \bar{\pi} : h \in \mathcal{H}\}$ . Since the tensor products related via permutations in  $\mathcal{H}$  represent exactly the same result, from now on we just need to consider  $\bar{\pi}$ . The action of  $g \in \mathcal{G}(S)$  induces an action on  $\pi(\Omega)$  via  $g \circ \pi(i) = \pi(g(i))$  for  $i \in \Omega$ . For a given coloring  $\pi$ , the *pointwise* stabilizer of it in  $\mathcal{G}(S)$  is denoted by  $\mathcal{G}_{(\pi)}(S) = \{g \in \mathcal{G}(S) : g \circ \pi(i) = \pi(i), \forall i \in \Omega\}$ . Then the number of colorings in  $\bar{\pi}$  is

$$|\bar{\pi}| = |\mathcal{G}(S)| / |\mathcal{G}_{(\pi)}(S)|. \quad (7)$$

Within this equivalent class  $\bar{\pi}$ , we can further classify the colorings based on the color patterns correspond to internal indices. For given  $\pi$ , we can write it as  $\pi = \pi^E \cup \pi^I$ , which distinguish the parts corresponding to external and internal indices. Consequently,  $\Omega$  can be partitioned into a disjoint union of  $\Omega_E(\pi)$  and  $\Omega_I(\pi)$ , which are supports of  $\pi^E$  and  $\pi^I$ , respectively. Then we can construct a set of unordered pairs by  $\varpi(\pi) = \{\theta(i, j) | \pi^I(i) = \pi^I(j), \forall i, j \in \Omega_I(\pi)\}$  where the head  $\theta$  specifies the type of internal indices. In terms of tensor products, we can call  $\varpi(\pi)$  as the contraction pattern of  $\pi$ . We say  $\pi_1, \pi_2 \in \bar{\pi}$  are equivalent, if  $\varpi(\pi_1) = \varpi(\pi_2)$ . It is easy to verify that this is indeed an equivalence relation on  $\bar{\pi}$ . In particular, the induced action of  $g$  on  $\varpi(\pi)$  can be defined as  $g \circ \theta(i, j) = \theta(g(i), g(j))$ . The *setwise* stabilizer of  $\varpi$  is denoted by  $\mathcal{G}_{\varpi(\pi)}(S) = \{g \in \mathcal{G}(S) : g \circ \varpi(\pi) = \varpi(\pi)\}$ . It is important to realize the group chain relation

$$\mathcal{G}(S) \geq \mathcal{G}_{\Omega_I(\pi)}(S) = \mathcal{G}_{\Omega_E(\pi)}(S) \geq \mathcal{G}_{\varpi(\pi)}(S) \triangleright \mathcal{G}_{(\Omega_E(\pi))}(S) \triangleright \mathcal{G}_{(\pi)}(S), \quad (8)$$

where  $\mathcal{G}_{\Omega_I(\pi)}(S)$  and  $\mathcal{G}_{\Omega_E(\pi)}(S)$  represents the setwise stabilizer of  $\Omega_I(\pi)$  and  $\Omega_E(\pi)$ , respectively. They are equal, because  $\forall g \in \mathcal{G}(S)$ ,  $g(i) \in \Omega_I(\pi), \forall i \in \Omega_I(\pi)$  is equivalent to  $g(i) \in \Omega_E(\pi), \forall i \in \Omega_E(\pi)$ . Eq. (8) shows that  $\mathcal{G}_{\varpi(\pi)}(S)$  is a subgroup of  $\mathcal{G}_{\Omega_I(\pi)}(S)$ . Note that the group  $\mathcal{G}_{(\Omega_E(\pi))}(S)$  is the pointwise stabilizer of  $\Omega_E(\pi)$  or equivalently  $\pi^E$  via the induced action  $g \circ \pi^E(i) = \pi^E(g(i))$ . Finally, the group  $\mathcal{G}_{(\pi)}(S)$  is a normal subgroup of  $\mathcal{G}_{\varpi(\pi)}(S)$ .



The importance of Eq. (8) is that we can further classify the colorings in  $\bar{\pi}$  according to the group chain using left coset decomposition. The first level decomposition  $\mathcal{G}(S) \geq \mathcal{G}_{\Omega_I(\pi)}(S) = \mathcal{G}_{\Omega_E(\pi)}(S)$  classifies  $\bar{\pi}$  into  $|\mathcal{G}(S)|/|\mathcal{G}_{\Omega_E(\pi)}(S)|$  classes that different classes have the different  $\Omega_E(\pi)$ . The next level decomposition  $\mathcal{G}_{\Omega_I(\pi)}(S) = \mathcal{G}_{\Omega_E(\pi)}(S) \geq \mathcal{G}_{\varpi(\pi)}(S)$  classifies the colorings with the same  $\Omega_E(\pi)$  according to their contraction pattern. In total, the nonequivalent classes with respect to the mapping  $\varpi$  can be obtained from the left coset decomposition

$$\mathcal{G}(S) = \bigcup_{r \in L} g_r \mathcal{G}_{\varpi(\pi)}(S), \quad (9)$$

where  $L$  is an index set and the set of left coset representatives  $g_r$  is denoted by  $\mathcal{L} = \{g_r : r \in L\}$ . Eq. (9) induces a partition of  $\bar{\pi}$  into disjoint classes with the same cardinality,

$$\bar{\pi} = \bigcup_{r \in L} \llbracket \pi_r \rrbracket, \quad \pi_r = g_r \circ \pi, \quad \llbracket \pi_r \rrbracket = \{g \circ \pi_r(\Omega) : g \in \mathcal{G}_{\varpi(\pi_r)}(S)\}, \quad (10)$$

where the class  $\llbracket \pi \rrbracket$ , i.e., the orbit of  $\pi$  under  $\mathcal{G}_{\varpi(\pi)}(S)$ , is the equivalent class of  $\pi$  under the equivalence relation  $\varpi(\pi_1) = \varpi(\pi_2)$ , and  $\mathcal{G}_{\varpi(\pi_r)}(S) = g_r \mathcal{G}_{\varpi(\pi)}(S) g_r^{-1}$ , because for  $g \in \mathcal{G}_{\varpi(\pi)}(S)$ ,  $g_r g g_r^{-1} \circ \varpi(\pi_r) = g_r g g_r^{-1} g_r \circ \varpi(\pi) = g_r \circ \varpi(\pi) = \varpi(\pi_r)$ . The number of nonequivalent classes  $\llbracket \pi_r \rrbracket$  is

$$|\mathcal{L}| = |\mathcal{G}(S)|/|\mathcal{G}_{\varpi(\pi)}(S)|. \quad (11)$$

The cardinality of  $\llbracket \pi_r \rrbracket$  is given by

$$|\llbracket \pi_r \rrbracket| = |\mathcal{G}_{\varpi(\pi)}(S)|/|\mathcal{G}_{(\pi)}(S)|, \quad \forall r \in L, \quad (12)$$

which is the order of the quotient group

$$\mathcal{G}_{\varpi(\pi)}(S)/\mathcal{G}_{(\pi)}(S) = \{g \mathcal{G}_{(\pi)}(S) : g \in \mathcal{G}_{\varpi(\pi)}(S)\} \quad (13)$$

Note the relation  $|\bar{\pi}| = |\mathcal{L}| \cdot |\llbracket \pi_r \rrbracket|$  is indeed fulfilled by Eqs. (7), (11), and (12).

Since  $\mathcal{G}_{\varpi(\pi)}(S)$  is a subgroup of  $\mathcal{G}_{\Omega_E(\pi)}(S)$ , its action leaves  $\Omega_E(\pi)$  invariant, and simply induces permutations on  $\Omega_E(\pi)$ . Let us look into this induced action in details.

**Theorem 2.** *The mapping from  $\mathcal{G}_{\varpi(\pi)}(S)$  to a permutation group on  $\Omega_E(\pi)$  denoted by  $\mathcal{G}(\Omega_E(\pi))$  defined via*

$$\phi : \mathcal{G}_{\varpi(\pi)}(S) \longrightarrow \mathcal{G}(\Omega_E(\pi)) \quad (14)$$

$$g \longmapsto \phi_g : \phi_g(i) = g(i), \forall i \in \Omega_E(\pi), \quad (15)$$

is a group homomorphism. The kernel of the mapping  $\text{Ker}\phi = \{g \in \mathcal{G}_{\varpi(\pi)}(S) : \phi_g = e\}$ , which is the pointwise stabilizer of  $\Omega_E(\pi)$  in  $\mathcal{G}_{\varpi(\pi)}(S)$ , simply gives the automorphism group of the contraction pattern  $\varpi(\pi)$ .

The proof is straightforward. By noting  $\forall i \in \Omega_E(\pi)$ ,  $\phi_{g_1 g_2}(i) = g_1 g_2(i) = g_1(g_2(i)) = \phi_{g_1} \phi_{g_2}(i)$ , thus  $\phi_{g_1 g_2} = \phi_{g_1} \phi_{g_2}$ . Besides,  $\phi_e = e$  and  $\phi_g^{-1} = \phi_{g^{-1}}$ . Therefore, the so-constructed  $\mathcal{G}(\Omega_E(\pi))$  is a permutation group acting on  $\Omega_E(\pi)$ , and  $\phi$  is a group homomorphism. According to the first isomorphism theorem of groups,  $\text{Ker}\phi$  is a normal subgroup of  $\mathcal{G}_{\varpi(\pi)}(S)$ , and  $\mathcal{G}(\Omega_E(\pi))$  is isomorphic to the quotient group  $\mathcal{G}_{\varpi(\pi)}(S)/\text{Ker}\phi$ . The support of  $\text{Ker}\phi$  is  $\Omega_I$  and  $g \circ \varpi(\pi) = \varpi(\pi)$  for  $g \in \text{Ker}\phi \triangleleft \mathcal{G}_{\varpi(\pi)}$ . Therefore,  $\text{Ker}\phi$  is the automorphism group of  $\varpi(\pi)$ .

For other contraction patterns  $\varpi(\pi_r)$  in Eq. (10), Eq. (15) leads to  $\mathcal{G}(\Omega_E(\pi_r)) = \phi(\mathcal{G}_{\varpi(\pi_r)}(E)) = g_r \mathcal{G}(\Omega_E(\pi)) g_r^{-1}$ . In particular, if  $g_r \in \mathcal{G}_{\Omega_E(\pi)}(S)$  then  $\mathcal{G}(\Omega_E(\pi_r)) = \mathcal{G}(\Omega_E(\pi))$ . In sum, different  $\mathcal{G}(\Omega_E(\pi_r))$  are isomorphic and if we relabel  $\Omega_E(\pi_r)$  by the same set of colors, i.e., the same set of external indices, then these groups induce exactly the same group on the colors (external indices). This is in fact the permutation symmetry group of tensor products in Question (iv).

With this theorem, we can partition  $\llbracket \pi_r \rrbracket$  in Eq. (10) by

$$\llbracket \pi_r \rrbracket = \llbracket \pi_r^E \rrbracket \times \llbracket \pi_r^I \rrbracket, \quad (16)$$

$$\llbracket \pi_r^E \rrbracket = \{g \circ \pi_r^E(\Omega_E) : g \in \mathcal{G}(\Omega_E(\pi_r))\}, \quad (17)$$

$$\llbracket \pi_r^I \rrbracket = \{g \circ \pi_r^I(\Omega_I) : g \in \text{Ker}\phi\}. \quad (18)$$

The cardinality of  $\llbracket \pi_r^E \rrbracket$  is

$$|\llbracket \pi_r^E \rrbracket| = |\mathcal{G}(\Omega_E(\pi_r))|, \quad (19)$$

since the colors in  $\pi_r^E$  are all different, while the cardinality of  $\llbracket \pi_r^I \rrbracket$ , which is the number of partial colorings  $\pi^I$  that have the same  $\varpi(\pi^I)$ , is given by

$$|\llbracket \pi_r^I \rrbracket| = |\text{Ker}\phi|/|\mathcal{G}_{(\pi)}(S)|, \quad (20)$$

which is the order of the quotient group  $\text{Ker}\phi/\mathcal{G}_{(\pi)}(S)$ . Note that  $|\llbracket \pi_r^E \rrbracket| \cdot |\llbracket \pi_r^I \rrbracket| = |\llbracket \pi_r \rrbracket|$  indeed recovers Eq. (12). Besides,  $|\mathcal{H}|/|\text{Ker}\phi/\mathcal{G}_{(\pi)}(S)|$  gives the number of classes that are nonequivalent with respect to  $\mathcal{G}(S)$  but equivalent with respect to  $\mathcal{H}$ , which means by

changing the colors for internals (or equivalently, relabeling the internal indices) these classes can be related. Essentially, they represent the same tensor product.

We note that given the permutation group  $\mathcal{G}(\Omega_E(\pi))$ , the classification of colorings of  $\Omega_E(\pi)$  with different types of external indices is again a combinatorial problem, in which the Polya's theorem can be applied directly. Therefore, now we can answer Question (i) by the above systematic classification of tensor products based on the hierarchy:

- (E1) the equivalence relation with respect to the factor set  $S$
- (E2) the equivalence relation with respect to both  $\mathcal{G}(S)$  on  $\Omega$  and  $\mathcal{H}$  on colors (internal indices),
- (E3) the equivalence relation with respect to  $\mathcal{G}(S)$  on  $\Omega$  but not to  $\mathcal{H}$ ,
- (E4) the equivalence relation with respect to the contraction pattern,
- (E5) the equivalence relation with respect to  $\mathcal{G}(\Omega_E)$  on  $\Omega_E$  and  $\text{Ker}\phi$  on  $\varpi(\pi)$ .

The counting of nonequivalent classes can be performed for (E2) based on de Bruijn's enumeration formula (Lemma 1), (E3) based on Polya's theorem (Lemma 1 with  $\mathcal{H} = \langle e \rangle$ ), (E4) with Eq. (9) for  $|\mathcal{L}|$ , (E5) with Eqs. (19) and (20) for  $|\llbracket \pi_r^E \rrbracket|$  and  $|\llbracket \pi_r^I \rrbracket|$ , respectively. This answers Question (ii).

Finally, we come to Question (iii), the possibility of selecting a representative for tensor products. It means to define a representative for a given coloring  $\pi$  from its  $(\mathcal{G}(S), \mathcal{H})$ -equivalent class  $\bar{\pi}$ . Having performed a clear classification of tensor products, it is straightforward to provide a natural definition of representatives.

**Theorem 3.** *Within a  $(\mathcal{G}(S), \mathcal{H})$ -equivalent class  $\bar{\pi}$ , we assume a priority of external indices over internal indices, then the following four conditions:*

- (C1)  $\Omega_E(\pi_{\text{canon}})$  is minimal in lexicographic order among  $\{\Omega_E(\pi_r) : r \in L\}$ , which is equivalent to  $\Omega_I(\pi_{\text{canon}})$  is maximal in lexicographic order among  $\{\Omega_I(\pi_r) : r \in L\}$ ,
- (C2)  $\varpi(\pi_{\text{canon}}^I)$  is minimal in lexicographic order among the colorings in  $\bar{\pi}$  satisfying (i),
- (C3)  $\pi_{\text{canon}}^E$  is minimal in lexicographic order of colors among the colorings that share the same  $\Omega_E$ ,

(C4)  $\pi_{\text{canon}}^I$  is minimal in lexicographic order of colors among all colorings that share the same  $\varpi(\pi_{\text{canon}}^I)$ ,

uniquely define a coloring  $\pi_{\text{canon}} = \mathcal{C}_\pi$  that constitutes a representative (canonical form) of  $\bar{\pi}$ , in the sense (1)  $\pi_{\text{canon}}$  is  $(\mathcal{G}(S), \mathcal{H})$ -equivalent to  $\pi$ , (2)  $\forall g \in \mathcal{G}(S), h \in \mathcal{H}, \mathcal{C}_{h\pi}(g\Omega) = \mathcal{C}_\pi(\Omega)$ . There are two special cases.

(S1) If there is no external index, then only (C2) and (C4) apply, because  $\Omega_I(\pi) = \Omega$ ,  $\Omega_E(\pi) = \emptyset$  for any  $\pi$ .

(S2) If there is no internal index at all, then only (C3) applies, in which case  $\varpi(\pi) = \emptyset$ ,  $\mathcal{G}_{\varpi(\pi)}(S) = \mathcal{G}(S) = \mathcal{G}(\Omega_E(\pi))$ , and  $\text{Ker}\phi = \langle e \rangle$ , because  $\Omega_E(\pi) = \Omega$ ,  $\Omega_I(\pi) = \emptyset$  for any  $\pi$ .

These conditions follow from the group chain (8) and the homomorphism  $\phi$  in Theorem 2. Basically, (C1) uniquely selects an equivalent class of colorings sharing the same  $\Omega_I$  (or  $\Omega_E$ ) according to the decomposition  $\mathcal{G}(S) \geq \mathcal{G}_{\Omega_I(\pi)}(S)$ , while (C2) uniquely selects  $\llbracket \pi_r \rrbracket$  within this class according to  $\mathcal{G}_{\Omega_I(\pi)}(S) \geq \mathcal{G}_{\varpi(\pi)}(S)$ . The condition (C3) selects an element of  $\llbracket \pi_r^E \rrbracket$ , while the condition (C4) selects an element of  $\llbracket \pi_r^I \rrbracket$ , which can be viewed as a labeling of internal indices for in accord with the contraction pattern  $\varpi(\pi)$ . The importance of representative is that  $\mathcal{C}_{\pi_1} = \mathcal{C}_{\pi_2}$  if and only if  $\pi_1$  and  $\pi_2$  are  $(\mathcal{G}(S), \mathcal{H})$ -equivalent. In terms of tensor products, this means the two tensors can be transformed between each other by permuting indices and relabeling internal indices properly. While Theorem 3 provides a well-defined representative, in practice we need an efficient algorithm to calculate the representative, in which the calculation  $\mathcal{G}_{\varpi(\pi)}(S)$  is the most crucial step. Before we step into the algorithm for the calculation of representatives, it is better to illustrate the above theoretical results with some concrete examples.

## B. Examples

Let us consider a simple tensor product formed by two tensors  $\mathbf{g}$  satisfying the symmetry

$$g_{pq,rs} = g_{qp,rs} = g_{pq,sr} = g_{rs,pq}, \quad (21)$$

that is  $\mathcal{G}(\mathbf{g}) = \langle (12), (34), (13)(24) \rangle$ . Thus, the symmetry for the factor set  $S = (\mathbf{g}, \mathbf{g})$  is  $\mathcal{G}(S) = (\mathcal{G}(\mathbf{g}) \times \mathcal{G}(\mathbf{g})) \rtimes \mathcal{S}_2(\mathbf{g}) = \langle (12), (34), (13)(24), (56), (78), (57)(68), (15)(26)(37)(48) \rangle$ ,

whose order is  $8 \times 8 \times 2 = 128$ . For the coloring type denoted by  $(n_{ext}\mathbf{e}, n_{int}\mathbf{i})$  ( $n_{ext}$  external indices and  $n_{int}$  internal indices), we have  $n_{ext} + n_{int} = D = 8$ ,  $n_c = n_{int}/2$ , and  $C = n_{ext} + n_c$ . Here, for simplicity, we assume that all the internal indices are of the same type such that  $\mathcal{H} = S_{n_c}$  (the symmetric group of degree  $n_c$ ), and also the external indices are of the same type. Then the number of  $(\mathcal{G}(S), \mathcal{H})$ -equivalent classes  $N_{gh}$  and the number of  $\mathcal{G}(S)$ -equivalent classes  $N_g$  can be calculated from Lemma 1. The results together with the number of different expressions  $N(n_{ext}, n_{int}) = D!/2^{n_c}$  ( $D = 8$  in this example) for each coloring type are summarized in Figure 1. We can see that there are totally  $8!(2 - 2^{-4}) = 78120$  different tensor expressions sharing the same factor set.

Now suppose we have an expression  $g_{i_1 i_2, i_3 i_3} g_{i_1 i_2, e_1 e_2}$  which corresponds to the coloring  $\pi = \{i_1, i_2, i_3, i_3, i_1, i_2, e_1, e_2\}$ , then following the lines of Sec. II A we will investigate the corresponding quantities  $\bar{\pi}$ ,  $\bar{\pi}$ ,  $\llbracket \pi \rrbracket$ ,  $\llbracket \pi^E \rrbracket$ ,  $\llbracket \pi^I \rrbracket$ , and the most important one  $\pi_{\text{canon}}$ . First, according to the definitions, we have

$$\pi^E = \{e_1, e_2\}, \quad (22)$$

$$\Omega_E(\pi) = \{7, 8\}, \quad (23)$$

$$\pi^I = \{i_1, i_2, i_3, i_3, i_1, i_2\}, \quad (24)$$

$$\Omega_I(\pi) = \{1, 2, 3, 4, 5, 6\}, \quad (25)$$

$$\varpi(\pi) = \{\theta(1, 5), \theta(2, 6), \theta(3, 4)\}. \quad (26)$$

and then the groups in Eq. (8) are calculated as

$$\mathcal{G}_{\Omega_I(\pi)}(S) = \langle (34), (56), (78), (13)(24) \rangle = \mathcal{G}_{\Omega_E(\pi)}(S), \quad (27)$$

$$\mathcal{G}_{\varpi(\pi)}(S) = \langle (34), (12)(56), (78) \rangle, \quad (28)$$

$$\mathcal{G}_{(\pi)}(S) = \langle (34) \rangle, \quad (29)$$

with

$$|\mathcal{G}_{\Omega_I(\pi)}(S)| = 32, \quad |\mathcal{G}_{\varpi(\pi)}(S)| = 8, \quad |\mathcal{G}_{(\pi)}(S)| = 2. \quad (30)$$

Then the number of expressions  $|\bar{\pi}|$  (7), the number of different contraction patterns  $|\mathcal{L}|$  (11), and the number of colorings in any class  $\llbracket \pi_r \rrbracket$  (12) are

$$|\bar{\pi}| = 128/2 = 64, \quad |\mathcal{L}| = 128/8 = 16, \quad |\llbracket \pi_r \rrbracket| = 8/2 = 4. \quad (31)$$

More specifically, the 16 classes  $[\pi_r]$  can be divided into  $|\mathcal{G}(S)|/|\mathcal{G}_{\Omega_I(\pi)}(S)| = 128/32 = 4$  groups based on their  $\Omega_I(\pi_r)$  (or equivalently  $\Omega_E(\pi_r)$ ). The sixteen  $\varpi(\pi_r)$  are illustrated pictorially in Figure 2, with the patterns in each row share the same  $\Omega_I(\pi_r)$ . It is clearly that the initial coloring  $\pi$  belongs to the 15-th class. According to Theorem 2, we have

$$\mathcal{G}(\Omega_E(\pi)) = \langle (78) \rangle, \quad \text{Ker}\phi = \langle (34), (12)(56) \rangle, \quad (32)$$

such that

$$[\pi^E] = \{\{e_1, e_2\}, \{e_2, e_1\}\}, \quad [\pi^I] = \{\{i_1, i_2, i_3, i_3, i_1, i_2\}, \{i_2, i_1, i_3, i_3, i_2, i_1\}\}, \quad (33)$$

and  $|\mathcal{H}|/|\text{Ker}\phi/\mathcal{G}_{(\pi)}(S)| = 3!/(4/2) = 3$  gives the number of classes that are nonequivalent with respect to  $\mathcal{G}(S)$  but equivalent with respect to  $\mathcal{H}$ , i.e.,  $|\bar{\pi}|/|\pi|$ .

Now we consider the selection of representatives. The condition (C1) implies the contraction patterns in the first row of Figure 2 should be selected, while the condition (C2) implies the first class is the choice, since  $\varpi(\pi_1)$  is lexicographically smaller than those for the other three class. The condition (C3) suggests  $\pi_{\text{canon}}^E = \{e_1, e_2\}$ , while the condition (C4) suggests  $\pi_{\text{canon}}^I = \{i_1, i_2, i_1, i_2, i_3, i_3\}$  because it is lexicographically minimal among the six possible ways for labeling internal indices in accord with  $\varpi(\pi_1)$ . Thus, we have  $\pi_{\text{canon}} = \{e_1, e_2, i_1, i_2, i_1, i_2, i_3, i_3\}$  and the corresponding canonical form of tensor product  $g_{e_1 e_2, i_1 i_2} g_{i_1 i_2, i_3 i_3} g_{i_1 i_2, e_1 e_2}$ .

Finally, it deserves to be emphasized that generally  $\mathcal{G}(\Omega_E(\pi))$  is not a subgroup of  $\mathcal{G}(S)$ . The following example illustrates this fact. For  $t_{e_1 e_2, i_1 i_2} h_{i_1 i_2}$  with  $\mathcal{G}(\mathbf{t}) = \langle (12)(34) \rangle$  and  $\mathcal{G}(\mathbf{h}) = \langle (12) \rangle$ , we have  $S = (\mathbf{t}, \mathbf{h})$ ,  $\mathcal{G}(S) = \langle (12)(34), (56) \rangle$ ,  $\pi = \{e_1, e_2, i_1, i_2, i_1, i_2\}$ ,  $\varpi(\pi) = \{\theta(3, 5), \theta(4, 6)\}$ . Then  $\mathcal{G}_{\Omega_I(\pi)}(S) = \mathcal{G}_{\Omega_E(\pi)}(S) = \mathcal{G}(S)$  and  $\mathcal{G}_{\varpi(\pi)}(S) = \langle (12)(34)(56) \rangle$  such that  $\mathcal{G}(\Omega_E(\pi)) = \langle (12) \rangle$  and  $\text{Ker}\phi = \langle e \rangle$ . Obviously,  $(12) \notin \mathcal{G}(S)$ , i.e.,  $\mathcal{G}(\Omega_E(\pi))$  is not a subgroup of  $\mathcal{G}(S)$ . Only through the homomorphism  $\phi$  (15), we can get the permutation symmetry group of  $\Omega_E(\pi)$ .

### III. ALGORITHMS

#### A. Traditional backtrack algorithm

As we can see from the above examples, to compute the canonical forms of tensor products 1, it is essential to be able to calculate  $\mathcal{G}_{\varpi(\pi)}(S)$ , which can determine all possible contraction

patterns via the left coset decomposition (9). The structure of  $\varpi(\pi)$  reveals it as a special combinatorial object. For this kind of problems, the backtrack searching<sup>13–15</sup> appears to be the only possible approach, which potentially involves searching through all of the elements of a permutation group, and hence the computational complexity is at least  $\mathcal{O}(|\mathcal{G}(S)|)$  in the worst case. To make it work in practice, it is crucial in designing such algorithms to find methods for skipping as many group elements as possible during the search. This is often refereed as pruning the search tree. The backtrack algorithms have also been used in problems including centralizers and normalizers of elements and subgroups, stabilizers of subsets of  $\Omega$ , and intersections of subgroups. For more applications, we refer the readers to Refs.<sup>14,15</sup>.

First, we show that in the special case (S2) with no internal indices, the canonical form can be efficiently obtained by a simple modification of the traditional backtrack. Given  $\Omega = \{1, 2, \dots, D\}$  and  $\mathcal{G} = \mathcal{G}(S)$ , denote the pointwise stabilizer of the  $i - 1$  first elements of  $\Omega$  ( $\Omega_{i-1}$ ) by  $\mathcal{G}^{[i]} \triangleq \mathcal{G}_{(\Omega_{i-1})}$ , then we will have the stabilizer chain,

$$\mathcal{G} = \mathcal{G}^{[1]} \geq \mathcal{G}^{[2]} \geq \dots \geq \mathcal{G}^{[D]} \geq \mathcal{G}^{[D+1]} = \langle e \rangle. \quad (34)$$

Let  $U^{(k)}$  be a left transversal (a set of left coset representatives) for  $\mathcal{G}^{[k+1]}$  in  $\mathcal{G}^{[k]}$ . Then every  $g \in \mathcal{G}$  can be uniquely decomposed into

$$g = u_{1,i_1} u_{2,i_2} \dots u_{D,i_D}, \quad u_{k,i_k} \in U^{(k)}, \quad (35)$$

which is referred as the Schreier-Sims representation<sup>13</sup> of  $\mathcal{G}$ . In particular,  $|\mathcal{G}| = \prod_{k=1}^D |U^{(k)}|$ . A backtracking procedure (depth-first transversal) can be used to run through the elements of  $\mathcal{G}$ , which results in an organization of the elements of  $\mathcal{G}$  as a search tree  $\mathcal{T}$  in accord with Eq. (35). More specifically, the root at level 0 labeled by empty represents  $\mathcal{G} = \mathcal{G}^{[1]}$ . The nodes at level  $k = 1$  represent the coset  $u_{1,i_1} \mathcal{G}^{[2]}$  ( $i_1 = 1, \dots, |U^{(1)}|$ ) labeled by  $(\gamma_{i_1})$ , where  $\gamma_{i_1} = u_{1,i_1}(1)$ . In general, every node at level  $k > 0$  is labeled with a sequence  $(\gamma_{i_1}, \dots, \gamma_{i_k}) \subseteq \Omega$ , which represents the coset  $u_{1,i_1} \dots u_{k,i_k} \mathcal{G}^{[k+1]}$ . The node  $(\gamma_{i_1}, \dots, \gamma_{i_k})$  has  $|U^{(k+1)}|$  children  $(\gamma_{i_1}, \dots, \gamma_{i_k}, \gamma_{i_{k+1}})$  for each  $\gamma_{i_{k+1}} \in (\Delta^{(k+1)})^g$  where  $\Delta^{(k+1)} = \{g(j) : g \in \mathcal{G}^{[k+1]}, j \in \Omega\}$  is the orbit of  $\mathcal{G}^{[k+1]}$  and  $g \in u_{1,i_1} \dots u_{k,i_k} \mathcal{G}^{[k+1]}$ , viz.,  $g$  is an arbitrary permutation fulfilling  $g(1, \dots, k) = (\gamma_{i_1}, \dots, \gamma_{i_k})$ . Therefore, at the level  $D$ , the leaves correspond to all the elements of  $\mathcal{G}$ . Each path from the root to a leaf represents a sequence of group elements  $u_{1,i_1}, (u_{1,i_1} u_{2,i_2}), (u_{1,i_1} u_{2,i_2} u_{3,i_3}), \dots, (u_{1,i_1} u_{2,i_2} \dots u_{D,i_D})$ , and from one element to the next element, one more image of point in  $\Omega$  is fixed.

Now suppose the coloring  $\pi(\Omega) = \{\pi(1), \pi(2), \dots, \pi(D)\}$  with all  $\pi(i)$  are different, then  $\mathcal{G}_{(\pi)}(\Omega) = \langle e \rangle$  and  $\mathcal{G}(\Omega(\pi)) = \mathcal{G}(S)$ . Especially, the elements in  $\llbracket \pi \rrbracket = \{g \circ \pi(\Omega) : g \in \mathcal{G}(S)\}$  have a one-to-one correspondence with the group elements  $g \in \mathcal{G}(S)$ . Then, the lexicographic ordering of colors  $\pi(i)$  leads to a natural ordering of the partial images like  $(\gamma_{i_1}, \dots, \gamma_{i_k})$  at the same level  $k$ . That is, we say  $\gamma_k = (\gamma_{i_1}, \dots, \gamma_{i_k}) < \gamma'_k = (\gamma_{i'_1}, \dots, \gamma_{i'_k})$  if  $\pi(\gamma_k) < \pi(\gamma'_k)$ . In particular, there is only one minimum  $\pi$  that satisfies (C3), i.e., only one minimum  $(\gamma_{i_1}, \dots, \gamma_{i_D})$  at the level  $D$  of the search tree  $\mathcal{T}$ . Most importantly, there is also a unique minimum at each level  $k$  of  $\mathcal{T}$ , which is simply obtained from the first  $k$  elements of the minimum at level  $k + 1$ . Therefore, we can prune the search tree by only retain the minimal partial image at each level. This suggests a modification of the traditional backtrack search, which is a uniform depth-first search, into a guided depth-first search. The order of the nodes to be visited in the next level is obtained by first taking a breadth-first search from the current node, and then comparing the partial images. This is illustrated in Figure 3 for  $g_{e_4 e_1, e_3 e_2}$  with  $\mathcal{G}(g) = \langle (12), (34), (13)(24) \rangle$ . The final result is given by  $\pi(\{2, 1, 4, 3\}) = \{e_1, e_4, e_2, e_3\}$ , i.e.,  $\mathcal{C}(g_{e_4 e_1, e_3 e_2}) = g_{e_1 e_4, e_2 e_3}$ . Note that a large parts of the search tree is pruned. In fact, for  $\mathcal{G}(S) = \mathcal{S}_D$ , the symmetric group of degree  $D$ , this procedure is similar to the selection sort for  $\pi$ , which has  $\mathcal{O}(D^2)$  time complexity instead of  $\mathcal{O}(|\mathcal{G}(S)|) = \mathcal{O}(D!)$ . In general, since at level  $k$  the number of point images that have been fixed are  $k$ , and the number of children for a node at level  $k$  will not exceed  $D - k$ , then we can conclude the number of nodes that will be visited will at most of  $\mathcal{O}(D^2)$ .

## B. Graphic representation of tensor products

Unlike for the special case (S2), in which there is a unique minimum at each level, in the presence of internal indices, several nodes may required to be explored when  $\text{Ker} \phi \neq \langle e \rangle$ . Moreover, in this case it is not natural to define an ordering for partial images, which is of significant importance in pruning the search tree. The pair structure of elements in  $\varpi(\pi)$  suggests that it is better to be viewed as a list of edges in a graph. Moreover, if there are external indices in  $\pi$ , then we enlarge  $\Omega$  to include  $D + 1$  natural numbers. The first element 1 is an auxiliary vertex, while the rest  $D$  elements correspond to those in the original  $\Omega$ . In this way, we can augment  $\varpi(\pi)$  with  $\theta(1, i)$  for  $i \in \Omega_E(\pi)$ , then the problem of finding the representative satisfying (C1)-(C4) can be solved in a single framework, viz., the



canonicalization of graphs, which is used in graph isomorphism problem. Before introducing the algorithm, we formalize the correspondence between tensor products and graphs more explicitly.

**Definition 6** (Graphic representation of tensor products). *An undirected graph is an ordered pair  $G = (\mathcal{V}, \mathcal{E})$ , where  $\mathcal{V}$  is a finite set of vertices or nodes, and  $\mathcal{E}$  is a set of unordered pairs of vertices called edges. For a tensor product of form (1), suppose its correspondent coloring is  $\pi$ , we associate  $\pi$  with a graph  $G$  composed of  $V = \Omega$  and  $\mathcal{E} = \{(1, j) : j \in \Omega_E(\pi)\} \cup \{(i, j) : i, j \in \Omega_I(\pi)\}$  for general cases. The graph in special cases (S1) and (S2) can be obtained correspondingly.*

Some examples for the introduced graphic representation of tensor products are provided in Figure 4. These include (a) a triple excitation operator  $T_{ijk}^{abc}$ , (b) a 5-site matrix product state, (c) the example studied in Sec. II B,  $g_{i_1 i_2, i_3 i_3} g_{i_1 i_2, e_1 e_2}$ , (d) a fully contracted term  $T_{ijkl}^{abcd} T_{lkij}^{cadb}$ . It is clear that these graphs have rather simple structures, namely except for the vertex '0' whose degree can be larger than one, the degree of other vertices is just one. Also, there are many disconnected components, if the number of internal indices is large. We can label the vertex set  $\mathcal{V}$  in accord with the factor set  $S$ , while there is some freedom to label the edge set  $\mathcal{E}$ . The labels for  $\mathcal{E}$  can be chosen as the information that fully specifies the index pair  $(i, j)$ . In this work, we label the pair by a pattern '*type*[*index*]', i.e., where '*type*' specifies the space of  $j$  (occupied, virtual, or others), and '*index*' can be the specific index appeared in the expressions or general unspecific indices, viz.,  $i$  for occupied,  $a$  for virtual, etc. In this convention, we have  $\theta(i, j) = \{(i, j), \text{type}[\text{index}]\}$  to specify an edge  $(i, j)$ . Depending on the '*index*' takes the value of specific or general index, we call the corresponding graph labeled or unlabeled. Then, for a given tensor product 1, or equivalently, a factor set  $S$  and coloring  $\pi$ , we have the following correspondence between graphs and algebraic quantities:

- (K1) The externally labeled, internally labeled graphs  $G^{EI}(\pi)$  with labeled edge set  $\mathcal{E}^{EI}(\pi)$  have a one-to-one correspondence with the full tensor expressions.
- (K2) The externally labeled, internally unlabeled graphs  $G^E(\pi)$  with labeled edge set  $\mathcal{E}^E(\pi)$  correspond to a combination of  $\pi_r^E$  with  $\llbracket \pi_r^I \rrbracket$ , which essentially represent the tensor products. The graphs shown in Figure 4 belong to the kind (K2).

- (K3) The externally unlabeled, internally labeled graphs  $G^I(\pi)$  with labeled edge set  $\mathcal{E}^I(\pi)$  correspond to a combination of  $\llbracket \pi_r^E \rrbracket$  with  $\pi_r^I$ .
- (K4) The externally unlabeled, internally unlabeled graphs  $G(\pi)$  with unlabeled edge set  $\mathcal{E}(\pi)$  correspond to  $\llbracket \pi_r \rrbracket$ , which can be termed as skeletons. Note that  $\mathcal{E}(\pi) = \{\theta(1, j) : j \in \Omega_E(\pi)\} \cup \varpi(\pi)$ . Examples of this kind of graph are just those shown in Figure 2 except for the absence of the vertex '0', in which by labeling external and internal indices it gives  $16 \times 2! \times 3! = 64 \times 3 = 192$  distinct expressions or distinct (K1) graphs.

In accord with these labeling schemes, we defined the corresponding label-preserved graph isomorphism as follows

**Definition 7** (Label-preserved graph isomorphism). *Two labeled graphs  $G_1 = (\mathcal{V}_1, \mathcal{E}_1)$  and  $G_2 = (\mathcal{V}_2, \mathcal{E}_2)$  are identical, i.e.,  $G_1 = G_2$ , if  $\mathcal{V}_1 = \mathcal{V}_2$  and  $\mathcal{E}_1 = \mathcal{E}_2$  in the sense that  $\theta(u, v) \in \mathcal{E}_1$  if and only if  $\theta(u, v) \in \mathcal{E}_2$ . Two graphs  $G_1 = (\mathcal{V}_1, \mathcal{E}_1)$  and  $G_2 = (\mathcal{V}_2, \mathcal{E}_2)$  are isomorphic under  $\mathcal{G}(S)$ , i.e.,  $G_1 \cong G_2$ , if  $\exists g \in \mathcal{G}(S)$  such that  $\theta(g(x), g(y)) \in \mathcal{E}_2$  if and only if  $\theta(x, y) \in \mathcal{E}_1$ , i.e.,  $g \circ \mathcal{E}_1 = \mathcal{E}_2$ .*

Consequently, the automorphism of the labeled graph is defined as

**Definition 8** (Automorphism of labeled graph). *If  $g \in \mathcal{G}(S)$ , such that  $g \circ G \triangleq (\mathcal{V}, g \circ \mathcal{E}) = G$ , i.e.,  $g \circ \mathcal{E} = \mathcal{E}$ , then  $g$  is an isomorphism from a graph  $G$  to itself, which is called an automorphism. The set of all automorphism of  $G$  in  $\mathcal{G}(S)$  forms the automorphism group  $\text{Aut}_{\mathcal{G}(S)}(G)$  of the labeled graph. We use the subscript to emphasize that the automorphism group is calculate from  $\mathcal{G}(S)$ . The action of  $\text{Aut}_{\mathcal{G}(S)}(G)$  on  $\mathcal{V}$  partitions  $\mathcal{V}$  into orbits, and induces an equivalence relation on  $\mathcal{V}$ :  $u$  and  $v$  are equivalent if and only if they are in the same orbit, i.e., there exists an automorphism  $g \in \text{Aut}_{\mathcal{G}(S)}(G)$  such that  $u^g = v$ .*

The canonical form of graph are defined as

**Definition 9** (Canonical form of graph). *The canonicalization is a mapping such that for all  $g \in \mathcal{G}(S)$  and labeled graph  $G$ , (C1)  $\mathcal{C}(G) \cong G$  and (C2)  $\mathcal{C}(G^g) = \mathcal{C}(G)$ . By the property (C2), which is called "label-invariance", the image  $\mathcal{C}(G)$ , called a canonical form, is a unique representative of its isomorphic class  $\{g \circ G : g \in \mathcal{G}(S)\}$ . The importance of canonical form is that  $G \cong G'$  if and only if  $\mathcal{C}(G) = \mathcal{C}(G')$ .*

For unlabeled graphs, the canonical form is usually defined by the graph having the smallest adjacency matrix among the isomorphic class of graphs. For our purpose, the natural definition is that the canonical graph has the smallest *sorted* labeled edge set  $\mathcal{E}(\pi)$ . The sorting can simply be based on the lexicographic order of  $\theta(i, j)$ . Clearly, such definition is the counterpart of that defined in Theorem 3. From these definitions, we can find the following connection:

**Theorem 4.** *Two tensor products are equivalent (or two colorings  $(S_1, \pi_1)$  and  $(S_2, \pi_2)$  are  $(\mathcal{G}(S), \mathcal{H})$ -equivalent), if and only if  $G^E(\pi_1) \cong G^E(\pi_2)$ . The computation of a representative for  $\pi$  defined in Theorem 3 is equivalent to the computation of canonical form for  $G^E(\pi)$ . In particular, we have  $\text{Aut}_{\mathcal{G}(S)}(G(\pi)) = \mathcal{G}_{\varpi(\pi)}(S)$  and  $\text{Aut}_{\mathcal{G}(S)}(G^E(\pi)) = \text{Ker}\phi$ , which follows from the fact that  $\text{Ker}\phi$  is the pointwise stabilizer of  $\Omega_E(\pi)$  in  $\mathcal{G}_{\varpi(\pi)}(S)$ .*

The importance of this theorem is that to compute the representative of tensor products, we can simply compute the canonical form of the corresponding externally labeled graph  $G^E(\pi)$ . If the permutation symmetry is also required, we can calculate from the quotient group  $\text{Aut}_{\mathcal{G}(S)}(G(\pi))/\text{Aut}_{\mathcal{G}(S)}(G^E(\pi))$  via the homomorphism  $\phi$  (15). It is also clear that the number of different tensors sharing the same factor set  $S$  and the same number of external indices equals the number of topologically distinct graphs of kind (K4) (distinct skeletons), i.e.,  $|\mathcal{G}(S)|/|\text{Aut}_{\mathcal{G}(S)}(G(\pi))|$ .

### C. Partition backtrack algorithm

Having established the correspondence between tensor products and labeled graphs, we present an algorithm to compute the canonical form of a graph and the generators of its automorphism group. This is mainly based on McKay's partition backtrack algorithm implemented in his `nauty` code<sup>16</sup>, which is currently the fastest algorithm for graph isomorphism problem, with some modifications to fit our special problem. The major difference between the graph isomorphism problem and our problem is that in the former case any permutation belongs to  $\mathcal{S}_n$  ( $n$  is the number of vertices) is allowed, while in our case the allowed permutations are given by  $\mathcal{G}(S)$ . Thus, our method for exhausting the elements of the group is similar to that used in traditional backtrack. But instead of working with partial images, we work with partitions of  $\mathcal{V}$ , for which a well-defined ordering exists naturally. This allows an

efficient pruning of the search tree. To further enhance the pruning, the local breadth-first search used in the above modified traditional backtrack is also used here. In case of large  $\text{Aut}_{\mathcal{G}(S)}(G^E(\pi))$ , which potentially enlarges the branching factors of the search tree, the idea of using automorphism to prune the search tree developed for the general isomorphism problem is used. The detailed account of the theoretical basis is omitted here for simplicity, and can be found in the original papers<sup>16</sup>.

## 1. Search tree

Most of the graph isomorphism algorithms including McKay's partition backtrack<sup>16</sup> employed the same "individualization-refinement" paradigm but differ in some details. The central quantity is the partition.

**Definition 10** (Partition). *An ordered partition of the set  $\mathcal{V}$  is a sequence of subsets  $\Pi = (\Pi_1, \Pi_2, \dots, \Pi_r)$ , such that  $\Pi_i \neq \emptyset, \mathcal{V} = \bigcup_{i=1}^r \Pi_i$ , and  $\Pi_i \cap \Pi_j = \emptyset$  for  $i \neq j$ . The sets  $\Pi_i$  are called cells of  $\Pi$ . A discrete ordered partition is an ordered partition with each cell being a singleton  $|\Pi_i| = 1$ .*

The following relation defines a partial order for the set of all ordered partitions  $\Pi(\mathcal{V})$ .

**Definition 11.** *For  $\Pi_1, \Pi_2 \in \Pi(\mathcal{V})$ , we say  $\Pi_2$  is finer than  $\Pi_1$ , denoted by  $\Pi_2 \preceq \Pi_1$ , if each cell of  $\Pi_1$  is a consecutive union of cells of  $\Pi_2$ .*

The individualization and refinement, which are used to construct the search tree, are defined as follows:

**Definition 12** (Individualization). *Let  $v \in \mathcal{V}$  belong to a non-singleton cell  $\Pi_i$  of an ordered partition  $\Pi$ , then  $\Pi \downarrow v = (\Pi_1, \dots, \Pi_{i-1}, \{v\}, \Pi_i \setminus \{v\}, \Pi_{i+1}, \dots, \Pi_r)$  denotes the partition obtained from  $\Pi$  by splitting  $\Pi_i$  into the cells  $\{v\}$  and the complement  $\Pi_i \setminus \{v\}$ . We call  $\Pi \downarrow v$  is obtained from  $\Pi$  by individualizing vertex  $v$ . Obviously, the relation  $\Pi \downarrow v \prec \Pi$  holds.*

**Definition 13** (Refinement). *A refinement of  $(G, \Pi)$  is a partition  $\mathcal{R}(G, \Pi)$  such that (i)  $\mathcal{R}(G, \Pi) = (G, \Pi')$  where  $\Pi' \preceq \Pi$ , (ii)  $\mathcal{R}$  preserves isomorphisms, which means if  $(G_1, \Pi_1) \cong (G_2, \Pi_2)$ , then  $\mathcal{R}(G_1, \Pi_1) \cong \mathcal{R}(G_2, \Pi_2)$ .*

Due to the simple structure of our graphs, the refinement used in the present work is quite straightforward.

**Definition 14** (Refinement based on connectivity). *For given  $(G, \Pi)$  and the associated subgroup  $\mathcal{G}_\Pi(S) = \{g \in \mathcal{G}(S) : \Pi_i^g = \Pi_i, \Pi_i \in \Pi\}$ , then the refinement  $\mathcal{R}(G, \Pi)$  is obtained by a repeated application of the following two operations until the partition is not changed:*

- (R1) *If the first vertex  $v$  in a non-singleton cell is stabilized by  $\mathcal{G}_\Pi(S)$ , then it can be singled out which leads to a new partition  $\Pi \downarrow v$ .*
- (R2) *Suppose  $(\{v_1\}, \{v_2\}, \dots, \{v_k\})$  are the first  $k$  singleton cells of  $\Pi$ , such that  $\Pi_{k+1}$  is a non-singleton cell, then we consider these singleton cells sequentially. For  $v_i$ , if its neighbor in  $G$  has more than one element (in our graph this can only be the case for  $v_1 = 1$  when there are external indices), then  $\Pi$  is returned. Suppose the neighbor of  $v_i$  is a single vertex  $u_i$ , if  $u_i$  is in a singleton cell, then we move to the next vertex  $v_{i+1}$ . On the other hand, suppose  $u_i$  is in a non-singleton cell, then if  $u_i$  is the first element or it can be moved to the first element by a permutation  $g \in \mathcal{G}_\Pi(S)$ , then we obtain a new partition  $\Pi^g \downarrow u_i$ , otherwise,  $\Pi$  is returned.*

The use of refinement is not essential for the final results, but is quite important for reducing the size of search tree. As an example, for the expression with only internal indices, without refinement the depth of the search tree is at most  $D$ , while with refinement the depth is at most  $D/2$ .

Note that the partition  $\Pi$  can be viewed as a coloring for  $\mathcal{V}$  (not to be confused with  $\pi$ ). A vertex colored graph is a pair  $(G, \Pi)$ , where  $G$  is a graph and  $\Pi$  is a coloring. For an initially-specified colored graph  $(G, \Pi_0)$ , the search tree  $\mathcal{T}(G, \Pi_0)$  is constructed by selecting the first non-singleton cell of  $\Pi_0$ , individualizing it in all possible ways, refining the new partition to new nodes  $R(G, \Pi, v)$ , and finally terminating when the leaves, i.e., the discrete partition, are reached. It is important to note that both the target cell selector, i.e., selecting the first non-singleton cell, and the refinement function are isomorphism invariant, which means if  $(G_1, \Pi_1) \cong (G_2, \Pi_2)$  then  $F(G_1, \Pi_1) \cong F(G_2, \Pi_2)$  with  $F$  being a function. The consequence is that two isomorphic graphs will have isomorphic search trees. The whole process for traversing the search tree is usually implemented by a backtrack searching algorithm. Note that the leaves have a one-to-one correspondence with the elements of  $\mathcal{G}(S)$ ,

because the discrete partitions are just images of  $\mathcal{V}$  under the action of  $\mathcal{G}(S)$ . Thus, after traversing the whole search tree, the canonical graph defined by having the smallest sorted labeled edge set  $\mathcal{E}(\pi)$  can be found. Also, the automorphisms can be found, because for two discrete partitions  $\Pi_1 = g_1(\mathcal{V})$  and  $\Pi_2 = g_2(\mathcal{V})$  with their correspondent  $\mathcal{E}(\pi)$  being identical, then  $g_1^{-1}g_2$  is an automorphism. In practice, the search tree  $\mathcal{T}(G, \Pi_0)$  can be extremely large, and is of practical use only if it can be pruned. In the following, we introduce two kinds of pruning, one is based on the non-discrete partitions and the other is based on the automorphism detected during the search.

## 2. Pruning with non-discrete partition

**Definition 15** (Position). *The position of a vertex  $v \in \mathcal{V}$  in an ordered partition  $\Pi$  is defined by  $p(v, \Pi) = 1 + \sum_{i=1}^{k-1} |\Pi_i|$  for  $v \in \Pi_k$ .*

With this definition and  $\mathcal{E}(\pi)$ , we can introduce a function of  $\Pi$ ,

$$\mathcal{E}(\pi, \Pi) = \{\theta(p(i, \Pi), p(j, \Pi)) : i, j \in \mathcal{V}, (i, j) \in \mathcal{E}(\pi)\}. \quad (36)$$

This function is actually the edge set of the quotient graph  $Q(G, \Pi) = \{\mathcal{V}', \mathcal{E}'\}$ , with  $\mathcal{V}' = \{p(v, \Pi) : v \in \mathcal{V}\}$  and  $\mathcal{E}' = \{\theta(p(u, \Pi), p(v, \Pi)) : \theta(u, v) \in \mathcal{E}(\pi)\}$ , if we consider the partition  $\Pi$  as an equivalence relation on  $\mathcal{V}$ , namely,  $u, v \in \mathcal{V}$  are called equivalent if they are in the same cell, i.e.,  $u, v \in \Pi_i$  for some  $i$ . The so-defined function  $\mathcal{E}(\pi, \Pi)$  has an important property

**Theorem 5.** *If  $\Pi_1 \prec \Pi_2$ , then  $\mathcal{E}(\pi, \Pi_1) > \mathcal{E}(\pi, \Pi_2)$ .*

This shows along a path from the root to a leaf, the value of  $\mathcal{E}(\pi, \Pi)$  is increasing. That is, the value of  $\mathcal{E}(\pi, \Pi)$  at a given node  $\nu$  is a lower bound for all the values  $\mathcal{E}(\pi, \Pi)$  of its descendent. Therefore, if at the node  $\nu$ , its corresponding  $\mathcal{E}(\pi, \Pi)$  is larger than the minimal  $\mathcal{E}_{\min}(\pi)$  we have currently, then the entire subtree  $\mathcal{T}(G, \Pi_0, \nu)$  can be pruned. To make this pruning more powerful, the breadth-first search is applied at the current node  $\nu$  to reorder the ordering of children to be visited in the depth-first search. This kind of lookahead strategy makes the pruning with  $\mathcal{E}(\pi, \Pi)$  to the maximum possible extent.

### 3. *Pruning with automorphism*

The pruning based on automorphism group is crucial for graphs having large automorphism groups. The basic idea is simple. At a given node  $\nu$  in  $\mathcal{T}(G, \Pi_0)$ , suppose we have an subgroup  $\Gamma$  of  $\text{Aut}_{G(S)}(G(\pi))$  at hand, then by applying its elements to the partition  $\Pi$ , we obtain several new partitions  $\Pi^g$ . If some  $\Pi^g$  has been visited before, then the entire subtree  $\mathcal{T}(G, \Pi_0, \nu)$  is the same as that visited subtree, such that it can be pruned. This pruning ensures that only the generators of the automorphism group are found, rather than all its elements during the backtrack searching. However, the computational cost of a naive implementation based on checking the action of every element of the subgroup will still scale as  $O(n!)$  when the order of the automorphism group scales as  $O(n!)$  for large  $n$ . This is the case for the kind of tensor expressions in Figure 4(d). Therefore, although the number of visited intermediate nodes are small, the time for checking will be very long. Therefore, how to use the automorphism detected during the search smartly should be also addressed.

The solution is also based on a classification of elements in  $\Gamma$ . Suppose the parent of the node  $(\nu, \Pi)$  is  $(\nu', \Pi')$  and the stabilizer of the ordered partition  $\Pi'$  is  $\Gamma_{\Pi'} = \{g \in \Gamma : g(\Pi'_i) = \Pi'_i, \Pi'_i \in \Pi'\}$ . Then we have the coset decomposition  $\Gamma = \bigcup_r g_r \Gamma_{\Pi'}$ . The meaning of this decomposition is clear. For  $g \in \Gamma_{\Pi'}$ , it only transforms the branches of the subtree  $\mathcal{T}(G, \Pi_0, \Pi')$ , while the left coset representative transform the entire subtree to another subtree. Now suppose  $\exists g \in \Gamma$  such that  $\Pi^g = g \circ \Pi$  has been visited before. Let the deepest common ancestor of  $\Pi$  and  $\Pi^g$  in the search tree  $\mathcal{T}(G, \Pi_0)$  be denoted by  $\Pi_0$ , and the subtrees contains  $\Pi$  and  $\Pi^g$  be  $\mathcal{T}$  and  $\mathcal{T}^g$ , respectively. Now there can be only two cases: (1)  $g \notin \Gamma_{\Pi'}$  or (2)  $g \in \Gamma_{\Pi'}$ . These two cases lead to two different kinds of pruning based on the detected automorphisms.

(P1) In the first case,  $g$  must have been found during the search of  $\mathcal{T}$  before visiting  $\Pi$ .

Actually, once such  $g$  is found at a leaf node of  $\mathcal{T}$ , we can trace back to  $\Pi_0$  and prune the entire subtree  $\mathcal{T}$ .

(P2) If such pruning has been employed, then at a given node, only the second case is left, in which instead of the full automorphism group  $\Gamma$ , only the subgroup  $\Gamma_{\Pi'}$  needs to be considered. Moreover, in this case, we only need to examine whether there is an element in the orbit of  $\Pi$  under the action of  $\Gamma_{\Pi'}$  that has been visited before. The

computational cost of such examination is negligible.

In sum, by taking these two economic pruning strategies, only the paths that lead to leaves corresponding to generators of the automorphism group are retained, while all the parts corresponding to a composition of generators can be pruned. Thus, even with a large automorphism group, in which case the pruning based on  $\mathcal{E}(\pi, \Pi)$  takes no effect, the search space can be significantly pruned. The pseudocode of our final algorithm is presented in Algorithm 1.



---

**Algorithm 1** Backtrack search for canonically labeling a graph and finding generators for its automorphism group

---

**Input:** Tensor expression (or its correspondent graph  $G$ ) and symmetry group  $\mathcal{G}$

**Output:** Canonical labeling and generators for  $\text{Aut}(G)$

```

1: function TOCANONICALFORM(Tensor expression, Group  $\mathcal{G}$ )
2:   Transform tensor expression to a graph  $G$ ,
3:   Initialize  $\Pi_0$ , the certificate  $\mathcal{E}_c = \mathcal{E}_{\max}$ , generating set  $K = \{ \}$ 
4:   Backtrack( $G, \Pi_0, \mathcal{G}$ )
5:   Back transform  $(G, \Pi_c)$  to expressions
6: end function
7: function BACKTRACK( $G, \Pi, \mathcal{G}$ )
8:   if  $\mathcal{E}(G, \Pi) > \mathcal{E}(G, \Pi_c)$  or  $(G, \Pi)$  is automorphic to a visited node then
9:     return
10:  end if
11:  if  $|V_i| = 1$  ( $i = 1, 2, \dots, n$ ) then
12:    if  $\mathcal{E}(G, \Pi) = \mathcal{E}(G, \Pi_c)$  then
13:      Found an automorphism  $\Pi^{-1}\Pi_c$  and append it to  $K$ 
14:      Go back to the deepest common ancestor to omit the entire subtree containing  $\Pi$ 
15:    else
16:      Find a smaller  $\mathcal{E}(G, \Pi)$  and update  $\Pi_c = \Pi$  and  $\mathcal{E}(G, \Pi_c) = \mathcal{E}(G, \Pi)$ 
17:    end if
18:  else
19:    Select the first non-singleton cell  $v$ 
20:    Compute  $\text{Stab}_{\mathcal{G}}(1)$  and the left coset representatives  $\mathcal{L}$ 
21:    for  $g \in \mathcal{L}$  do
22:      Compute  $\Pi^g$ , split it to  $\Pi^g \downarrow v_k$ , and refine  $R(G, \Pi^g \downarrow v_k)$ 
23:    end for
24:    Sort the refinements  $\{R(G, \Pi^g \downarrow v_k)\}$  by  $\mathcal{E}(G, R(G, \Pi^g))$ 
25:    for  $\Pi_i \in \{R(G, \Pi^g \downarrow v_k)\}$  do
26:      Backtrack( $G, \Pi_i, \text{Stab}_{\mathcal{G}}(1)$ )
27:    end for
28:  end if
29: end function

```

---

## IV. RESULTS AND DISCUSSIONS

In this section, we start with some simple tensor products to show how the above algorithm works, and then provide results for more complicated expressions.

**Example 1:**  $I_{ij}^{ab} = t_i^a t_j^b$ .

This example gives  $S = (t, t)$ ,  $\pi = (a, i, b, j)$ ,  $\mathcal{G}(S) = \mathcal{S}_2(\mathbf{t}) = \langle (13)(24) \rangle$ . The corresponding graph is constructed via  $\mathcal{V} = \{1, 2, 3, 4, 5\}$  and

$$\mathcal{E}^E(\pi) = \{\{(1, 2), \text{vir}[a]\}, \{(1, 3), \text{occ}[i]\}, \{(1, 4), \text{vir}[b]\}, \{(1, 5), \text{occ}[j]\}\}.$$

To compute the permutation symmetry group, we use the externally unlabeled edge set

$$\mathcal{E}(\pi) = \{\{(1, 2), \text{vir}[a]\}, \{(1, 3), \text{occ}[i]\}, \{(1, 4), \text{vir}[a]\}, \{(1, 5), \text{occ}[i]\}\}.$$

The initial partition is  $\Pi_0 = \{\{1\}, \{2, 3, 4, 5\}\}$  and accordingly the group  $\mathcal{G}(S)$  is replaced by  $\mathcal{G}(S) = \langle (24)(35) \rangle$ . Then the above algorithm gives the visiting sequence

$$\begin{aligned} \Pi_0 &= \{\{1\}, \{2, 3, 4, 5\}\} \\ \rightarrow \Pi_1 &= \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}\} \\ \rightarrow \Pi_2 &= \{\{1\}, \{4\}, \{5\}, \{2\}, \{3\}\}. \end{aligned} \tag{37}$$

The stabilizer of the first element 2 in the non-singleton cell in  $\Pi_0$  is the trivial group  $\langle e \rangle$ , and the left coset representatives are just  $\{e, (24)(35)\}$ . The partition  $\Pi_1$  is obtained from  $\Pi_0 = \{\{1\}, \{2, 3, 4, 5\}\}$  by first individualizing 2,  $\Pi_0 \downarrow 2 = \{\{1\}, \{2\}, \{3, 4, 5\}\}$ , and then immediately  $\mathcal{R}(G, \Pi_0 \downarrow 2) = \Pi_1$  since the subgroup of  $\mathcal{G}(S)$  that stabilizes  $\Pi_0 \downarrow 2$  is the trivial group. Then the backtrack search proceeds to the image of the next left coset representative  $\Pi \downarrow 4 = \{\{1\}, \{4\}, \{5, 2, 3\}\}$ , which leads to  $\mathcal{R}(\Pi \downarrow 4) = \Pi_2$ . Since  $\mathcal{E}(\pi, \Pi_1) = \mathcal{E}(\pi, \Pi_2)$  for the two discrete partition  $\Pi_1$  and  $\Pi_2$ , then we have the automorphism group  $\text{Aut}_{\mathcal{G}(S)}(G(\pi)) = \langle (24)(35) \rangle$ , which essentially shows  $I_{ij}^{ab} = I_{ji}^{ba}$  through the homomorphism  $\phi(15)$ .

**Example 2:**  $\langle ij || ab \rangle t_i^a t_j^b$ .

This tensor product without any external index appears in the coupled-cluster energy expression. In our convention, we have  $S = (\bar{\mathbf{g}}, \mathbf{t}_1, \mathbf{t}_1)$  and  $\pi = (i, j, a, b, a, i, b, j)$ . The group of the antisymmetrized integral is  $\mathcal{G}(\bar{\mathbf{g}}) = \langle (12), (34), (13)(24) \rangle$  and according  $\mathcal{G}(S) = \mathcal{G}(\bar{\mathbf{g}}) \times \mathcal{S}_2(\mathbf{t}_1) = \langle (12), (34), (13)(24), (57)(68) \rangle$ , which is of order 16. The initial partition is

$\Pi_0 = \{\{1, 2, 3, 4\}, \{5, 6, 7, 8\}\}$ . The breadth-first search produces the following ordering for the children in increasing order of  $\mathcal{E}(\pi, \Pi_i)$ ,

$$\begin{aligned}
\Pi_1 &= \mathcal{R}(\Pi_0 \downarrow 3) = \mathcal{R}(\{\{3\}, \{4, 1, 2\}, \{5, 6, 7, 8\}\}) \\
&= \{\{3\}, \{4\}, \{1\}, \{2\}, \{5\}, \{6\}, \{7\}, \{8\}\} \\
\Pi_2 &= \mathcal{R}(\Pi_0 \downarrow 4) = \mathcal{R}(\{\{4\}, \{3, 1, 2\}, \{5, 6, 7, 8\}\}) \\
&= \{\{4\}, \{3\}, \{2\}, \{1\}, \{7\}, \{8\}, \{5\}, \{6\}\} \\
\Pi_3 &= \mathcal{R}(\Pi_0 \downarrow 1) = \mathcal{R}(\{\{1\}, \{2, 3, 4\}, \{5, 6, 7, 8\}\}) \\
&= \{\{1\}, \{2\}, \{3\}, \{4\}, \{5\}, \{6\}, \{7\}, \{8\}\} \\
\Pi_4 &= \mathcal{R}(\Pi_0 \downarrow 2) = \mathcal{R}(\{\{2\}, \{1, 3, 4\}, \{5, 6, 7, 8\}\}) \\
&= \{\{2\}, \{1\}, \{4\}, \{3\}, \{7\}, \{8\}, \{5\}, \{6\}\}.
\end{aligned}$$

The visit of  $\Pi_1$  and  $\Pi_2$  found an automorphism (12)(34)(57)(68) as  $\mathcal{E}(\pi, \Pi_1) = \mathcal{E}(\pi, \Pi_2)$ . The visit of  $\Pi_3$  found that it produces an edge list larger than the minimal edge list found so far, thus both  $\Pi_3$  and the partition after it, viz.,  $\Pi_4$ , can be pruned, since  $\mathcal{E}(\pi, \Pi_4) \geq \mathcal{E}(\pi, \Pi_3) > \mathcal{E}(\pi, \Pi_1)$  due to the reordering after the breadth-first scan. Finally, based on  $\mathcal{E}(\pi, \Pi_1) = \{\{(1, 5), \text{vir}[a]\}, \{(2, 7), \text{vir}[a]\}, \{(3, 6), \text{occ}[i]\}, \{(4, 8), \text{occ}[i]\}\}$ , we can relabel the tensor product as  $g_{a_1 a_2, i_1 i_2} t_{i_1}^{a_1} t_{i_2}^{a_2}$ , which is just the canonical form. It deserves to be mentioned that in the diagrammatic technique for coupled-cluster theory, the inverse of the order of the automorphism group is just the weight factor associated with the diagram<sup>17</sup>, which will be added to the expression when taking summations over internal indices. In the diagrammatic coupled cluster theory, the two  $\mathbf{t}_1$  vertices in this example are called equivalent vertices.

**Example 3:**  $\langle ij || ab \rangle t_{ij}^{ab}$ .

This tensor product also appears in the coupled-cluster energy expression. Similar to the second example, we have  $S = (\bar{\mathbf{g}}, \mathbf{t}_2)$  and  $\pi = (i, j, a, b, a, b, i, j)$ . The group is  $\mathcal{G}(S) = \mathcal{G}(\bar{\mathbf{g}}) \times \mathcal{G}(\mathbf{t}_2) = \langle (12), (34), (13)(24), (56), (78) \rangle$ , which is of order 32. This example is slightly more complicated than the previous two, and is also very typical concerning with the pruning based on automorphism. The practical search tree is shown in Figure 5. It is seen that once the automorphism (34)(78) is found, the entire subtree at  $\{\{4\}, \{3\}, \{1, 2\}, \{5, 6, 7, 8\}\}$  is pruned, since it will be mapped to the visited subtree at  $\{\{3\}, \{4\}, \{1, 2\}, \{5, 6, 7, 8\}\}$  by (34)(78). Besides, the entire subtree at  $\{\{2\}, \{1\}, \{3, 4\}, \{6\}, \{5\}, \{7, 8\}\}$  is also completely pruned, since the automorphism (12)(56) will map it to the visited subtree at

$\{\{1\}, \{2\}, \{3, 4\}, \{5\}, \{6\}, \{7, 8\}\}$ . The final automorphism group is found as  $\langle (12)(56), (34)(78) \rangle$ , which is of order 4. In the diagrammatic coupled-cluster theory, when summed over internal indices, this will contribute the weight factor  $1/4$ .

**Example 4:**  $t_{kji}^{cba}$ .

This example illustrates how the partition backtrack works for the tensor products with only external indices. The coloring is  $\pi = \{c, b, a, k, j, i\}$ . The initial partition is  $\Pi_0 = \{\{1\}, \{2, 3, 4, 5, 6, 7\}\}$  and  $\mathcal{G}(S) = \mathcal{G}(\mathbf{t}_3) = \langle (23), (234), (56), (567) \rangle$ , which is of order 36. After the following four steps

$$\begin{aligned}\Pi_1 &= \{\{1\}, \{4\}, \{2, 3, 5, 6, 7\}\} \\ \Pi_2 &= \{\{1\}, \{4\}, \{3\}, \{2\}, \{5, 6, 7\}\} \\ \Pi_3 &= \{\{1\}, \{4\}, \{3\}, \{2\}, \{7\}, \{5, 6\}\} \\ \Pi_4 &= \{\{1\}, \{4\}, \{3\}, \{2\}, \{7\}, \{6\}, \{5\}\}\end{aligned}$$

we arrive at  $\pi_{\text{canon}} = \{a, b, c, i, j, k\}$  and the corresponding canonical form of tensor  $t_{ijk}^{abc}$ . This shows in the case that there is no internal index, the partition backtrack algorithm behaves as a selection sort for  $\{c, b, a\}$  and  $\{k, j, i\}$ .

**Example 5:**  $A_{i_{\sigma(1)}i_{\sigma(2)}\dots i_{\sigma(n)}}A_{i_{\tau(1)}i_{\tau(2)}\dots i_{\tau(n)}}$  with  $\sigma, \tau \in \mathcal{S}_n$  and  $\mathcal{G}(\mathbf{A}) = \mathcal{S}_n$ .

This example is used to illustrate the computational cost of the partition backtrack. To this end, the search tree for  $n = 5$  is shown in Figure 6. The red points represent the partitions that are not pruned, the blue ones represent the partitions that are pruned by (P2), and the gray ones represent the partitions that are pruned by (P1) due to the detection of an automorphism. We can count the number of these different partitions, which gives  $N_{\text{red}} = (n+1)(n+2)/2$ ,  $N_{\text{blue}} = (n-2)(n-1)/2$ , and  $N_{\text{gray}} = \sum_{i=1}^{n-1} i(i+1)/2 = (n-1)n(n+1)/6$ . Therefore, the size of the search tree is  $N_{\text{tot}} = N_{\text{red}} + N_{\text{blue}} + N_{\text{gray}} = \frac{1}{6}n^3 + n^2 - \frac{1}{6}n + 2$ . Note that the automorphism group in this case is the same as  $\mathcal{G}(S) = (\mathcal{G}(\mathbf{A}) \times \mathcal{G}(\mathbf{A})) \times \mathcal{S}_2(\mathbf{A})$  is of order  $2(n!)$ , which grows extremely fast. For  $n = 20$ , the order is  $2 \times 20! \approx 4.9 \times 10^{18}$ , while the size of the search tree is  $N_{\text{tot}} = 1732$ , which is still tractable. In view of the algorithm 1, the computational cost is roughly proportional to  $N_{\text{tot}}$ . Therefore, we can conclude that the computational cost is of  $\mathcal{O}(n^3)$  for this example. In comparison, as mentioned in Sec. III A, the canonicalization of the externally labeled expression  $A_{e_{\sigma(1)}e_{\sigma(2)}\dots e_{\sigma(n)}}$ , which does not possess any automorphism, is much simpler, with computational cost of  $\mathcal{O}(n^2)$ . The tensor product in this example is more difficult due to the large automorphism group, which implies

that there are many partitions with the same edge sets. In other words, from the perspective of optimization, the final discrete partitions with minimal edge sets are saddle points in the landscape of  $\mathcal{E}(\pi, \Pi)$ , because they are maximums along the path from the root to them, and minimums among partitions at the same level. The landscape for minimization of the edge set in this example has many equivalent parts that can be related by automorphism. This greatly increases the difficulty in finding the saddle points compared with the former case, in which there is exactly one such point.

## V. CONCLUSION AND OUTLOOK

The present work provides a full classification of tensor expressions by means of equivalence relation and group chain. We give an analytic construction of symmetry group of the resulted tensor, a definition of canonical form of tensor based on its graph counterpart. We also provide a practical backtrack algorithm to calculate the canonical form and symmetry group, without which the previous elegant theoretical definitions are useless.

A few remarks can be drawn on the connection of the present general graphic representations with the traditional diagrams used in quantum chemistry<sup>17</sup>. The Goldstone diagrams can be viewed as a specific adaptation of the present diagrams to the case with only particle-hole indices and the symmetry group is often the symmetric group, such that the corresponding weight of diagrams are simply  $2^{n_{loop}}$  where  $n_{loop}$  represents the number of closed loops. The Hugenholtz diagrams, which is obtained via a homomorphic mapping of the Goldstone diagrams, is similar to our graph in shape, but differ in that the lines from one vertex are not distinguished by labeling.

The canonicalization for expression in MBPT with canonical orbitals is also possible by using hypergraph (set system), in which an edge can connect any number of vertices.  $E = \{\{v_1, v_2, v_3\}, \{v_4, v_5\}, \{v_6\}\}$ . In this case the introduction of the external vertex is not required, since a vertex can solely represent an edge by itself, e.g.,  $\{v_6\}$  in this example. The algorithm presented in this paper still works as long as a well-defined ordering is provided to compare  $E$ . This can be simply achieved, for instance, by comparing two edge sets first based on the number of members in a hyper-edge  $n$ , and then the type of edges, and finally the ordering of the sorted vertices. Accordingly, there are some minor modifications of the classification theory. The enumeration polynomial is the same, but now the terms with

powers other than 1 or 2 are also relevant. The concept of contraction pattern is extended to hyper-edge set  $\varpi_n(\pi)$ , where the hyper-edges containing  $n$  vertices are collected into a set (pattern). Consequently, the group chain is also extended by a chain of  $\mathcal{G}_{\varpi_n(\pi)}$ .

The theory and algorithm introduced in this paper can be applied in automatic derivation and simplification. The automorphism group  $\text{Ker}\phi$  and the permutation group  $\mathcal{G}(\mathcal{V}_E(\pi))$  contain significant information about the tensor products. In particular, when augmented with more general definition of tensor symmetry, more interesting information can be extracted from these groups. We will present such generalization in future.

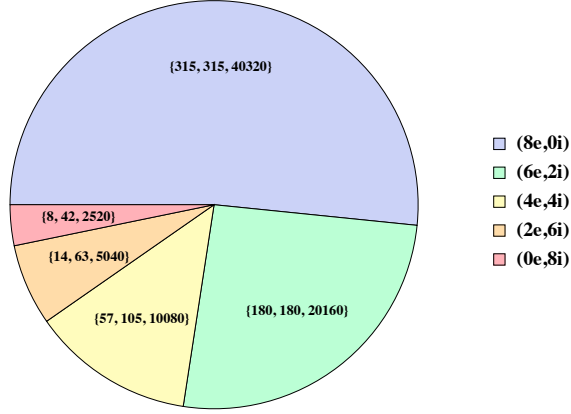


FIG. 1. Classification of different tensor expressions sharing the same factor set  $S = (\mathbf{g}, \mathbf{g})$  with  $\mathcal{G}(\mathbf{g}) = \langle (12), (34), (13)(24) \rangle$ . The pair  $(n_{ext}\mathbf{e}, n_{int}\mathbf{i})$  represents the coloring type with  $n_{ext}$  external indices and  $n_{int}$  internal indices, The triple  $\{N_{gh}, N_g, N(n_{ext}, n_{int})\}$  is a collection of three numbers:  $N_{gh}$  the number of  $(\mathcal{G}(S), \mathcal{H})$ -equivalent classes,  $N_g$  the number of  $\mathcal{G}(S)$ -equivalent classes, and  $N(n_{ext}, n_{int})$  the total number of different expressions.

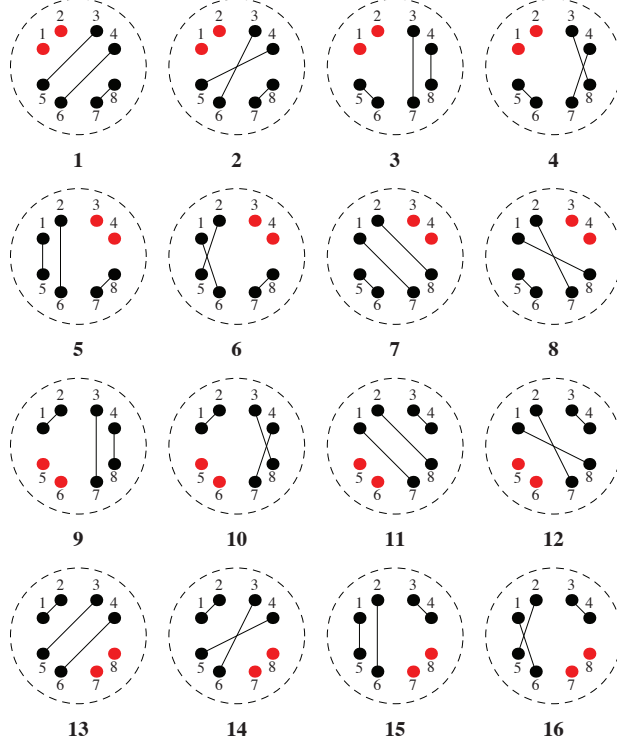


FIG. 2. The sixteen different contraction patterns  $\varpi(\pi_r)$ .

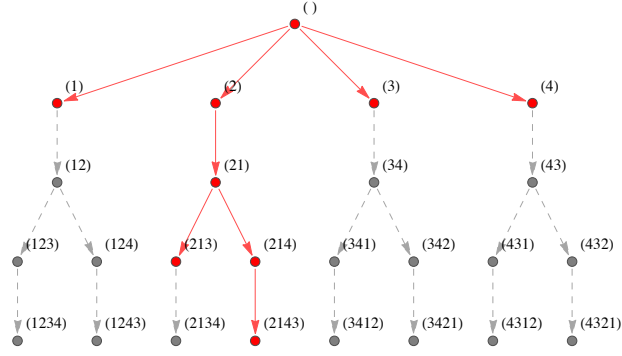


FIG. 3. Search tree of the modified traditional backtrack in finding representative for  $g_{e_4 e_1, e_3 e_2}$  with  $\mathcal{G}(\mathbf{g}) = \langle (12), (34), (13)(24) \rangle$ . The red parts represent the visited nodes and the actual search path, while all the gray parts are pruned. The final result is  $\mathcal{C}(g_{e_4 e_1, e_3 e_2}) = g_{e_1 e_4, e_2 e_3}$ .



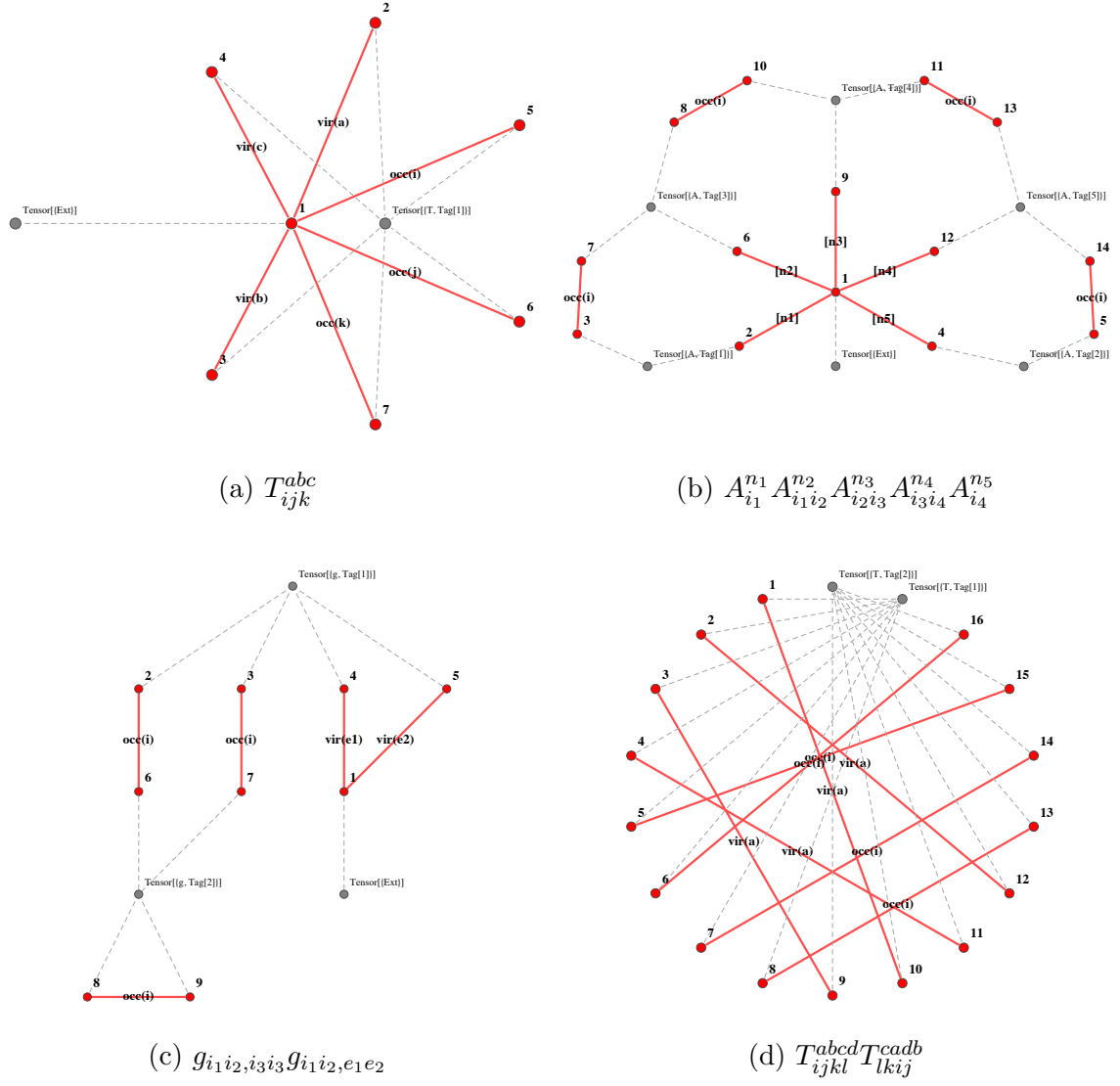


FIG. 4. Graphic representation of tensor expressions. The red parts form the graphs, while the gray parts reveal the origin of vertices.

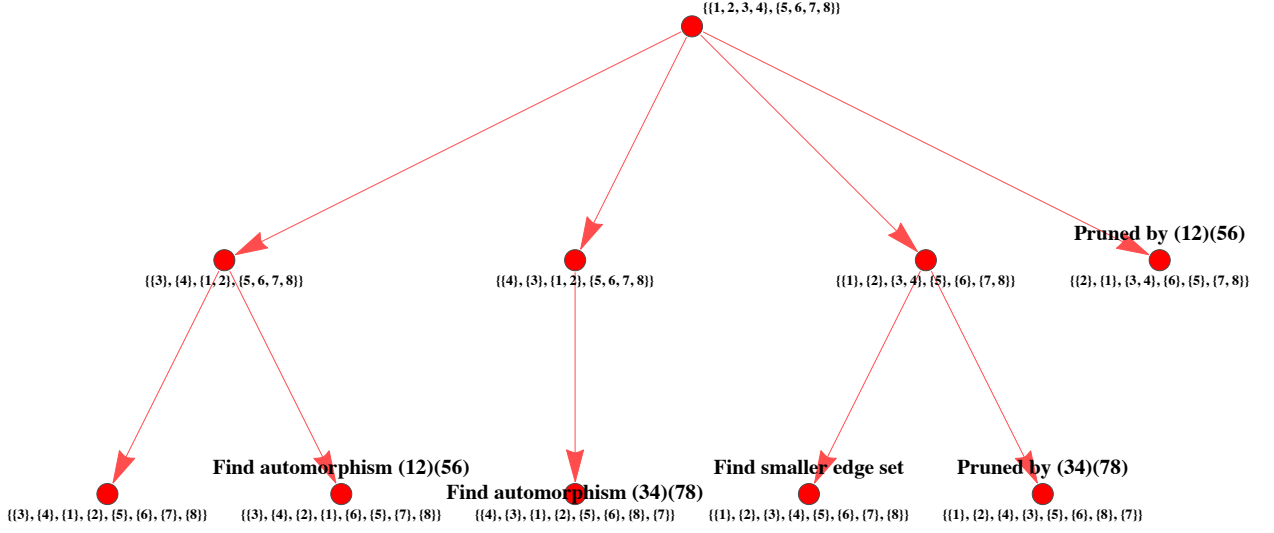


FIG. 5. Search tree for  $\langle ij || ab \rangle t_{ij}^{ab}$ .

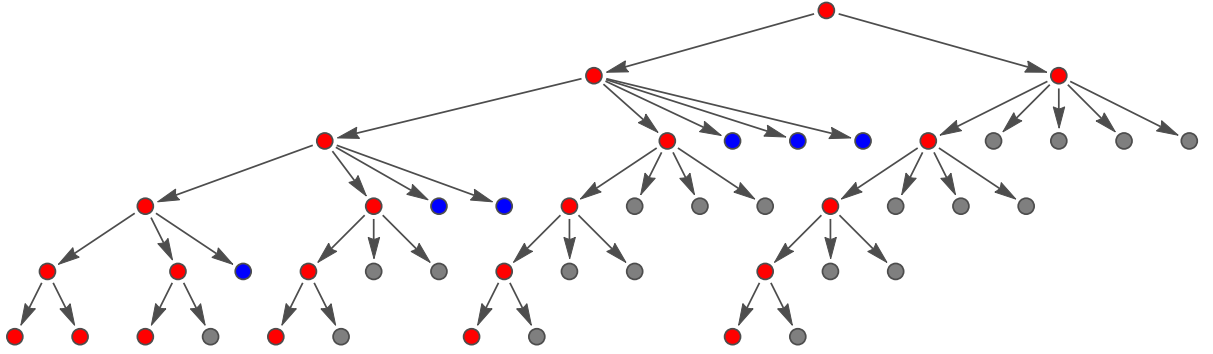


FIG. 6. Search tree for  $A_{i_1 i_2 i_3 i_4 i_5} A_{i_1 i_2 i_3 i_4 i_5}$  with  $\mathcal{G}(\mathbf{A}) = S_5$ .

## REFERENCES

- <sup>1</sup>A. Szabo and N. S. Ostlund, *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory* (Dover, 1996).
- <sup>2</sup>C. L. Janssen and H. F. Schaefer III, *Theor Chim Acta.* **79**, 1 (1991).
- <sup>3</sup>S. Hirata, *J. Phys. Chem. A* **107**, 9887 (2003).
- <sup>4</sup>R. Portugal, *J. Phys. A: Math. Gen.* **32** 7779 (1999).
- <sup>5</sup>R. Portugal and B. F. Svaiter, *Group-theoretic Approach for Symbolic Tensor Manipulation: I. Free Indices*, arXiv:math-ph/0107031v1.
- <sup>6</sup>L. R. U. Manssur and R. Portugal, *Group-theoretic Approach for Symbolic Tensor Manipulation: II. Dummy Indices*, arXiv:math-ph/0107032v1.
- <sup>7</sup>L. R. U. Manssur, R. Portugal, and B. F. Svaiter, *Internat. J. Modern Phys. C* **13** 859 (2002).
- <sup>8</sup>G. Butler, *On Computing Double Coset Representatives in Permutation Groups*, *Computational Group Theory*, ed. M.D. Atkinson, Academic Press (1984) pp. 283-290.
- <sup>9</sup>G. Pólya, *Acta Math.* **68**, 145 (1937).
- <sup>10</sup>J. H. Redfield, *Amer. J. Math.* **49**, no. 3, 433 (1927).
- <sup>11</sup>N. G. de Bruijn, *J. Combin. Theory* **2**, 418 (1967).
- <sup>12</sup>N. G. de Bruijn, *Nieuw Arch. Wisk. (3)* **19**, 89 (1971).
- <sup>13</sup>D. L. Kreher and D. R. Stinson. *Combinatorial Algorithms: generation, enumeration, and search*, Discrete Mathematics and Applications (CRC Press, 1999).
- <sup>14</sup>Á. Seress, *Permutation Group Algorithms* (Cambridge University Press, 2003).
- <sup>15</sup>D. F. Holt, B. Eick, and E. A. O'Brien. *Handbook of Computational Group Theory*, Discrete Mathematics and Applications (Chapman & Hall/CRC, 2005).
- <sup>16</sup>B. D. McKay and A. Piperno, *Practical graph isomorphism, II*, arXiv:1301.1493v1.
- <sup>17</sup>J. Paldus, J. Čížek, *Adv. Quantum Chem.* **9**, 105 (1975).